Research Article

GCWOAS2: Multiobjective Task Scheduling Strategy Based on Gaussian Cloud-Whale Optimization in Cloud Computing

Lina Ni ^{1,2} Xiaoting Sun ¹ Xincheng Li ¹ and Jinquan Zhang ^{1,2}

¹College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China

²Key Laboratory of the Ministry of Education for Embedded System and Service Computing, Tongji University, Shanghai 201804, China

Correspondence should be addressed to Jinquan Zhang; tjzhangjinquan@126.com

Received 26 January 2021; Accepted 30 May 2021; Published 10 June 2021

Academic Editor: Carmen De Maio

Copyright © 2021 Lina Ni et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

An important challenge facing cloud computing is how to correctly and effectively handle and serve millions of users' requests. Efficient task scheduling in cloud computing can intuitively affect the resource configuration and operating cost of the entire system. However, task and resource scheduling in a cloud computing environment is an NP-hard problem. In this paper, we propose a three-layer scheduling model based on whale-Gaussian cloud. In the second layer of the model, a whale optimization strategy based on the Gaussian cloud model (GCWOAS2) is used for multiobjective task scheduling in a cloud computing which is to minimize the completion time of the task via effectively utilizing the virtual machine resources and to keep the load balancing of each virtual machine, reducing the operating cost of the system. In the GCWOAS2 strategy, an opposition-based learning mechanism is first used to initialize the scheduling strategy to generate the optimal scheduling scheme. Then, an adaptive mobility factor is proposed to dynamically expand the search range. The whale optimization algorithm based on the Gaussian cloud model is proposed to enhance the randomness of search. Finally, a multiobjective task scheduling algorithm based on Gaussian whale-cloud optimization (GCWOA) is presented, so that the entire scheduling strategy can not only expand the search range but also jump out of the local maximum and obtain the global optimal scheduling strategy. Experimental results show that compared with other existing metaheuristic algorithms, our strategy can not only shorten the task completion time but also balance the load of virtual machine resources, and at the same time, it also has a better performance in resource utilization.

1. Introduction

With the rapid development of Internet of Things and big data technology, cloud computing [1] also occupies the most important position in business interconnection [2]. Users do not need to master the underlying application technology to use various resources in the cloud platform on demand [3, 4]. Cloud computing provides users with a more convenient way to use, and more and more people have noticed the cloud platform. Cloud computing integrates and manages a large number of idle resources through the Internet and manages and redistributes them. At the same time, users use the cloud platform to send tasks to the data center of the cloud terminal. Therefore, cloud computing systems are faced with hundreds of millions or even billions of jobs that

need to be scheduled every day. The number of tasks is very large, and each job occupies a certain amount of cloud system resources [5]. Cloud resources are heterogeneous, dynamic, and limited. Therefore, under the premise of satisfying user QoS, it is the field of cloud computing to efficiently schedule massive tasks and allocate cloud resources reasonably so that cloud tasks occupy as few cloud resources as possible [6]. The challenging issue is also a common concern of scholars today.

The scheduling problem of cloud computing is mainly divided into three layers: the first is scheduling virtual resources for user applications [7], the second is scheduling virtual resources (virtual machines) to physical resources, and the third is the scheduling and landing of physical resources. These kinds of scheduling have multiple different

goals to be optimized at each layer, so the scheduling problem in cloud computing is extremely complex, and it is considered as an NP-hard problem. In this article, we focus on studying the first layer of cloud computing scheduling, which is creating a VM to allocate resources reasonably for tasks sent by users [8]. At this layer, tasks are placed in virtual machines for execution. In the process of task execution, scheduling technology is particularly important. It affects the operating efficiency of the entire system, user service quality, system load balancing, system energy consumption, etc. [9–11]. Therefore, at this level, we must design an appropriate scheduling strategy to solve the above problems.

In the cloud computing environment, many forms of scheduling methods have been adopted, including priority-based scheduling [12–14], cluster-based scheduling [15, 16], QoS-based scheduling [17, 18], multiprocessor-based scheduling [19], and heuristic algorithm-based scheduling [20]. We can use exhaustive scheduling algorithms to meet scheduling requirements when the task request is small and the scale of cloud computing is also small. However, as the scale of the problem becomes larger, the dimensions and the variables we need to optimize are increasing, and exhaustive algorithms will be destroyed by dimensions. At this time, heuristic algorithms began to receive more and more attention. The heuristic algorithm is more efficient than the blind search algorithm. Although the optimal solution cannot be obtained for the NP-hard problem, the heuristic algorithm can approach the optimal solution infinitely and obtain a relatively optimal solution. In recent years, various heuristic algorithms have been used in scheduling problems in various fields, including production scheduling and cloud computing task scheduling. Summarizing the previous research, most of them still use the EC algorithm based on the overall [21], for example, the cloud computing task scheduling which is based on genetic algorithm [22], ant colony algorithm [23], and particle swarm optimization algorithm [24, 25]. These algorithms can effectively use the information exchange in the group to achieve the purpose of global search.

Among EC algorithms, the whale optimization algorithm (WOA) [26] is a new heuristic algorithm that simulates the predation behavior of humpback whales. Compared with other heuristic optimization algorithms, the whale optimization algorithm (WOA) uses random and search agents to simulate the hunting behavior of humpback whales and the spirals to simulate the attack behavior of the humpback whales in these two ways in order to find the best position of the humpback whale. This algorithm has a simple search strategy, strong search ability, and fast convergence speed. When applied to the cloud computing task scheduling process, it can quickly converge to obtain the most suitable scheduling plan. However, the convergence accuracy of WOA is very low, and the algorithm can easily fall into a local optimum. In the entire task scheduling process, it is very likely that the local optimal solution will be found, and the resulting scheduling plan is not the best plan among all choices.

The cloud model [27] is an uncertain transformation model that deals with qualitative concepts and quantitative descriptions, which includes forward cloud generators and reverse cloud generators. It has universality, randomness, and fuzziness and can realize the conversion of fuzzy concepts to specific data. The Gaussian cloud model can reflect the randomness of the concept by using the Gaussian distribution and the universality of the Gaussian membership function. However, the normal Gaussian cloud model is not very effective in finding the optimal solution of the problem, so applying it to the task scheduling process cannot find the most suitable scheduling scheme. Inspired by bionics and cloud generators, in this article, we consider combining the whale optimization algorithm (WOA) with the Gaussian cloud model and a multiobjective task scheduling strategy based on Gaussian cloud-whale optimization (GCWOAS2) is proposed for task scheduling in cloud computing.

In this article, our main purpose is to find a reasonable mapping between tasks and virtual machines to reduce execution time and reduce costs while meeting user's QoS. Therefore, we designed a three-layer scheduling model to achieve multiobjective scheduling and optimization. In the GCWOAS2 strategy, taking into account the blindness of the randomly initialized population, the opposition-based learning (OBL) mechanism [26] is introduced. In this way, the fitness of this individual and the opposing individual can be obtained, and individuals with less fitness can be selected from the two to form a new population, thereby obtaining a better population than randomly generated individuals, and further obtaining a better scheduling plan in the initial stage. In addition, we also introduce an adaptive learning factor to make the search step in this strategy adaptive. In this way, the search process can dynamically change according to the environment, which also enables the user's task to search for the VM in a larger range during the scheduling process. In addition, the Gaussian cloud model is introduced in the whale's swimming process, and a multiobjective task scheduling algorithm based on the Gaussian whale-cloud optimization-GCWOA algorithm is proposed. This algorithm not only has the randomness and universality of Gaussian cloud model, but also the random search function can also be applied to the WOA algorithm under the guidance of qualitative knowledge. In the process of multiobjective optimization, the scheduling process is more random, the task will be easier to find a suitable virtual machine and generate a suitable scheduling plan, and it can also make the algorithm jump out of the local optimum as soon as possible to avoid premature convergence.

In this paper, we study the multiobjective task scheduling problem based on the Gaussian whale-cloud model in cloud computing. The main contributions are summarized as follows:

- (1) A three-layer architecture model of whale-Gaussian cloud scheduling is proposed, which are user task layer, task scheduling layer, and data center layer to describe the entire process of task scheduling.

- (2) In the population initialization process, an initialization population strategy based on the opposition-based learning mechanism is proposed to generate the scheduling plan, so that the initialized scheduling plan is closer to the optimal plan. Simulation experiments show that each individual in the new population is closer to the optimal solution.
- (3) An adaptive adjustment factor based on dynamic changes is proposed to dynamically adjust the search step size to make the task search for VMs in a wider range in the scheduling process. Experiments have proved that this strategy can get more solutions than before.
- (4) A whale optimization algorithm based on the Gaussian cloud model is proposed, which makes the scheduling process more random and fuzzier, can make the scheduling strategy jump out of the local optimum, and enhances the optimization ability, with significant effects.
- (5) Finally, a multiobjective task scheduling algorithm based on the Gaussian whale-cloud optimization-GCWOA algorithm is proposed, and the GCWOA algorithm is used in the GCWOAS2 scheduling strategy, which can effectively combine all the advantages of the above contributions.

The other parts of this article are arranged as follows: in the second part, we study the related work. In the third part, we describe the task scheduling model. In the fourth part, the basic knowledge is introduced. In the fifth part, we propose our own method. The sixth part compares and evaluates our models. And the seventh part draws a summary.

2. Related Work

The heterogeneity and dynamics of cloud resources make the scheduling strategy for effectively allocating resources to tasks particularly complicated. In addition, the CPU running rate and bandwidth of virtual machines and hosts in the cloud are different, which will cause scheduling problems becoming more difficult.

2.1. Heuristic Algorithm Task Scheduling. To make more reasonable and effective use of resources in the cloud, metaheuristic algorithms [28] have begun to be gradually applied to cloud computing and are developing rapidly. Scholars at home and abroad began to propose many task scheduling schemes based on metaheuristic algorithms, for example, scheduling algorithms based on GA, PSO, and ACO.

Xiong et al. [29] analyzed the scheduling problem on the cloud data center, combined Johnson's algorithm with the genetic algorithm, designed a two-stage task scheduling, and optimized the makespan of each virtual machine. In order to reasonably arrange the problem of virtual machine placement under data-intensive services, in [30], Wang et al. proposed an improved particle swarm optimization algorithm, which weighs the energy consumption of the task and

the global QOS guarantee. Zhao et al. [31] combined Bayes' theorem and clustering process to find the best cluster of hosts, which improves system throughput and achieves overall load balancing.

Ding et al. [32] proposed a two-stage energy-saving cloud computing method based on Q-learning to reduce task response time and improve resource utilization. In [33], Sanaj and Joe Prathap proposed a MAP reduction framework and GA-WOA algorithm to schedule tasks in the cloud, and obtained a more efficient task scheduler. In [34], Dhinesh Babu and Venkata Krishna proposed a heuristic algorithm based on bee behavior to maximize throughput and achieve load balancing of virtual machines. Chen et al. [35] proposed an improved WOA algorithm to improve the convergence speed and accuracy in the scheduling process and improve resource utilization.

However, the scheduling focus of these methods is on the optimal solution of a single goal, and the overall situation cannot be considered. Although this example achieves the best results, it will lead to the effects of other examples are not very well, so that the results have great limitations.

2.2. Multiobjective Task Scheduling. With the application of cloud computing in the industrial Internet, single-objective optimization can no longer meet industrial needs, and multiobjective optimization has begun to attract more and more attention. Many scholars have conducted a lot of research in the context of multiobjective scheduling. The current research on multiobjective optimization mainly includes task completion time, QoS quality, operating cost, and energy consumption [36].

Hosseinzadeh et al. [37] conducted extensive research on multiobjective optimization tasks and workflow scheduling, classified and compared the multiobjective optimization algorithm schemes, and obtained the advantages and limitations of each scheme. Mohammadzadeh et al. [38] proposed a hybrid multiobjective greedy ALO algorithm based on chaos theory. The algorithm is optimized for multi-objective workflow scheduling problems to reduce manufacturing time and operating costs and improve throughput.

In [39], Xu et al. proposed a new multiobjective task scheduling model to obtain a suitable task allocation strategy. Peng et al. [40] proposed an online resource scheduling framework based on the deep Q-network (DQN) algorithm. The purpose is to resolve the conflict between cloud service providers who want to minimize energy costs and customers seeking to optimized services reducing energy consumption and task manufacturing. Huang et al. [41] considered multiobjective planning in cloud computing task scheduling and proposed a multiobjective particle swarm optimization to increase the search capability of the feasible solution space in the algorithm and enhance the global and local search capabilities.

In evolutionary computation, balancing the diversity and convergence of the population for multiobjective evolutionary algorithms (MOEAs) is one of the most challenging topics [42]. In the project scheduling problem, in

order to solve the problem of limited multiskill tasks, Wang and Zheng [43] proposed a knowledge-guided multi-objective fruit fly optimization algorithm (MOFOA) to minimize the manufacturing cycle and total cost. In Ali et al.'s study [44], to minimize manufacturing time and total cost in a foggy cloud environment, a multiobjective task scheduling optimization model based on the discrete non-dominated sorting genetic algorithm II (DNSGA-II) is proposed. Kumari et al. [45] designed a super heuristic resource scheduling based on the multiobjective particle swarm and genetic algorithm to improve resource utilization and throughput and reduce manufacturing time.

Although the above method has good performance in multiobjective scheduling, it has insufficient advantages in the convergence speed and convergence accuracy of the algorithm, and further optimization is needed.

In order to solve the above problems, in this paper we propose a multiobjective task scheduling strategy based on Gaussian cloud-whale optimization (GCWOAS2). A good scheduling strategy can effectively reduce the response time of tasks, meet the needs of users with different constraint levels, and further improve resource utilization and reduce energy consumption. So far, we have not found any research on applying the WOA algorithm to cloud task scheduling. Therefore, we apply the GCWOAS2 strategy proposed to cloud task scheduling in this paper. Experiments have proved that this can effectively reduce task execution time and balance the load.

3. System Model

In cloud computing, scheduling strategies can directly affect the utilization efficiency of resources at all levels and the operating costs of service providers, so the quality of task scheduling becomes the key to the smooth and convenient completion of production operations [46].

3.1. Whale-Gaussian Cloud Scheduling Model. In this article, we designed a whale-Gaussian cloud scheduling model, as shown in Figure 1. The model is composed of three layers, which are user task layer, task scheduling layer, and data center layer. The details in the system model are as follows:

- (1) User task layer: in this layer, after the user sends a task, a series of workloads are split into subtasks and placed in the task waiting queue so that the tasks can be scheduled in the next layer.
- (2) Task scheduling layer: in this layer, task monitors and resource managers feed back the remaining tasks and resource information to the control center, and the control center feeds back the information to the nodes in the cloud environment. Tasks are scheduled on the computing nodes according to our proposed GCMWOA algorithm. After the scheduling is completed, the task and resource information is fed back to the control center. In this way, nodes in the cloud can cooperate and communicate to complete the user's task request.

- (3) Data center layer: in this layer, the virtual machine information is sent to the server, and the scheduling and landing of physical resources are completed in the server.

The demand information submitted by the user is transformed into a set of subtasks at the task level. These subtasks enter the task waiting queue. At the scheduling layer, tasks begin to find suitable resource nodes in the cloud environment. The model can clearly divide all processes of task scheduling and make the scheduling of each part more intuitive. However, the scheduling strategy and scheduling algorithm we proposed are mainly concentrated on the second layer of the model-task scheduling layer. In this process, we assume that all tasks are independent of each other.

When we use algorithms to schedule tasks to virtual machines for execution, since it is unknown which virtual machine the task is suitable for running in, many solutions will be randomly generated when the algorithm initializes the population, so we propose to introduce the opposition-based learning mechanism to the initialization. In the population, when the population is initialized, the generated solution will be closer to the optimal solution. We have introduced an adaptive dynamic adjustment factor, so that our proposed algorithm will be able to search for as many tasks and virtual machine matching schemes as possible, which is conducive to obtaining the optimal solution. Since the algorithm may save the local suboptimal solution as the optimal solution in the iterative process, in order to avoid this problem, we introduce the Gaussian cloud model so that the algorithm can jump out of the local optimum and find the optimal solution.

3.2. Model Definition. To conveniently express the task scheduling process we designed, we give the following definitions:

Definition 1. Tasks and VM set: suppose there is a set of virtual machine resources recorded as $\{M_1, M_2, \dots, M_m\}$ and $N > M$. Any computing resource VM can be described as $\{id, cpu, mem, bw\}$, respectively, representing the virtual machine number, the execution speed, memory, and bandwidth of the virtual machine, and the unit is generally MIPS. The task is described as $\{id, length\}$, which, respectively, represent the task number and task length.

Definition 2. Mapping matrix A between tasks and VMs: each task in Definition 1 can only run on one virtual machine. One virtual machine can handle multiple tasks. All tasks are independent of each other. The correspondence between virtual machines is represented by a matrix as follows:

$$A_{nm} = \begin{bmatrix} a_{11}, a_{12}, \dots, a_{1m} \\ a_{21}, a_{22}, \dots, a_{2m} \\ \dots \\ a_{n1}, a_{n2}, \dots, a_{nm} \end{bmatrix}. \quad (1)$$

FIGURE 1: Whale-Gaussian cloud scheduling model architecture.

We use a_{ij} to represent the mapping relationship between tasks and virtual machines. When $a_{ij} = 1$, it means that the third task is executed on the fourth virtual machine. For each i , we stipulate $\sum_{j=1}^m a_{ij} = 1$.

Definition 3. Time function: in this article, the length of the task set and the running speed of the virtual machine are known. Our goal is to optimize task completion time, balance virtual machine load, and reduce operating costs. The time can be calculated according to formula (2), as follows:

$$ET(i, j) = \frac{\text{length}_i}{\text{vcpu}_j}, \quad 1 \leq i \leq n, 1 \leq j \leq m, \quad (2)$$

where $ET(i, j)$ represents the time it takes for task i to execute on virtual machine j , length_i represents the length of task i , and vcpu_j represents the running speed of virtual machine j . The task completion time is the maximum time

for the virtual machine to complete the task, expressed by formulas:

$$\text{time}_j = \sum_{i=1}^m a_{ij} ET(i, j), \quad (3)$$

$$f_1 = \max\{\text{time}_1, \text{time}_2, \dots, \text{time}_m\}, \quad (4)$$

where time_j represents the sum of the execution time of all tasks on the virtual machine j ($1 \leq j \leq m$) and f_1 represents the maximum value of the task execution time on the virtual machine.

Definition 4. Load cost function: load balancing means that the completion time of the virtual machine is basically the same as the total completion time of the task. We define the load balancing function as shown in the following formula:

$$f_2 = a_{ij} \sqrt{\frac{\sum_{j=1}^m (vt_j - \text{vavg}_j)^2}{m}}. \quad (5)$$

In formula (5), vt_j represents the running time of the task on virtual machine j , that is, the load on virtual machine j . vavg_j represents the average execution time of tasks executed on virtual machine j , that is, the average load on virtual machine j . f_2 is the smaller load of the virtual machine which is more balanced.

Definition 5. Operational cost function: the cost spent on a virtual machine is calculated by using the CPU, memory, and bandwidth of the virtual machine in this article. We define the cost function as shown in

$$f_3 = a_{ij} \sum_{j=1}^m (v_{\text{cpu}j} \times P_1 + v_{\text{bw}j} \times P_2 + v_{\text{ram}j} \times P_3). \quad (6)$$

In formula (6), $v_{\text{cpu}j}$ represents the CPU operating rate of the j -th virtual machine, $v_{\text{bw}j}$ represents the virtual machine bandwidth, and $v_{\text{ram}j}$ represents the memory size of the j -th virtual machine.

In order to more accurately converge and get the optimal solution, we use the min-max method to normalize the matrix. This method is used for normalization because the numerical proportions of the three targets f_1 , f_2 , and f_3 are quite different. In this case, the search for the optimal solution will be biased and the target with the largest proportion will be biased. It will affect the optimization accuracy.

Definition 6. Multiobjective function: express the above three objective functions as F_1 , F_2 , and F_3 , and normalize as shown in the following formulas:

$$F_1 = \frac{f_1}{\max(f_1)}, \quad (7)$$

$$F_2 = \frac{f_2}{\max(f_2)}, \quad (8)$$

$$F_3 = \frac{f_3}{\max(f_3)}. \quad (9)$$

Different cloud computing systems have different performance requirements for target tasks. Therefore, in this article, we convert the above three objective function utilization weights from multiobjective scheduling to single-objective scheduling. The final objective function becomes

$$F_{\text{obj}} = \min\{w_1 F_1 + w_2 F_2 + w_3 F_3\}. \quad (10)$$

In formula (8), the weight w can be adjusted according to the proportion according to the actual application requirements of the system. For example, if we pay more attention to optimizing time in the whole process, we can take $w_1 = 0.5$, $w_2 = 0.25$, and $w_3 = 0.25$. At this time, a large number of tasks will be allocated to the VM with relatively high CPU running speed and relatively large memory to save time.

From the overall optimization of the algorithm, in order to obtain the smallest F_{obj} , the larger the value of w_i , the smaller the value of corresponding F_i , which will result in different scheduling. Whether the task is allocated to a VM with a larger CPU or to a VM with a larger memory and resource bandwidth will be affected by the value of w_i . This configuration of using weights to influence the optimization target will speed up the convergence speed of the algorithm. Here, we do not consider the extreme cases of $w_1 = 1$, $w_2 = 0$, and $w_3 = 0$. This setting transforms the problem into single-objective scheduling.

Definition 7. Optimization objective function: in this article, the three objectives to be optimized are equally important, so we set $w_1 = w_2 = w_3$. Based on this situation, we get the formula

$$F_{\text{obj}} = \min\left\{\frac{1}{3}F_1 + \frac{1}{3}F_2 + \frac{1}{3}F_3\right\}. \quad (11)$$

In this article, our main optimization goals are task execution time, virtual machine load, and the cost of running all virtual machines. In the process of task scheduling, the commonly used method to reduce task execution time is to place the task in the virtual machine with the fastest running speed, that is, the largest CPU value, but this will cause the load of the virtual machine with a large CPU value to be too high. Other virtual machines with small CPU values are idle, resulting in unbalanced load, which in turn affects operating costs. This kind of operation is more efficient in terms of time, but it sacrifices the load of the virtual machine and also affects the cost. Therefore, we propose a multiobjective scheduling in this article. Our strategy can take into account the time and load issues at the same time, so that all objectives are optimized.

4. Preliminaries

In this part, we give the main ideas and algorithms of the whale optimization algorithm and Gaussian cloud model and use them as the basis for cloud computing task scheduling based on the GCMWOA algorithm.

4.1. Whale Optimization Algorithm (WOA). Whale optimization algorithm (WOA) [26] is an algorithm proposed based on the behavior of whales in hunting prey. In this algorithm, the position of each whale represents a mapping between tasks and virtual machines in scheduling. In D -dimensional space, the position of the whale is $X = \{x_1, x_2, \dots, x_D\}$. In the process of hunting, whales will have two behaviors: one is to surround the prey and the other is to drive the prey with a bubble net. In the process of swimming, the whale will randomly choose these two behaviors to hunt down prey.

The probability of choosing whether to surround the prey or the bubble net to trap the prey is the same when the whale swims, that is, $P(\text{enclose}) = P(\text{bubble net}) = 0.5$. When a whale chooses to surround its prey, it will choose to move to the optimal position or to a random whale position.

4.1.1. Surround Prey. When choosing to move to the optimal position,

$$\vec{D} = \vec{C} \times \vec{X}_b(t) - \vec{X}(t), \quad (12)$$

$$\vec{X}(t+1) = \vec{X}_b(t) - \vec{A} \times \vec{D}, \quad (13)$$

where $\vec{D}$ represents the enclosing step size, t represents the current algebra, $\vec{A}$ and $\vec{C}$ are coefficient vectors, and $\vec{X}_b$ represents the optimal position vector in the solution space, and the calculation formulas for coefficient vectors $\vec{A}$ and $\vec{C}$ are shown in

$$\vec{A} = 2\vec{a} \times \vec{r}_1 - \vec{a}, \quad (14)$$

$$\vec{C} = 2 \times \vec{r}_2. \quad (15)$$

In each iteration process, $\vec{r}_1$ and $\vec{r}_2$ are random vectors between $[0, 1]$. As the convergence factor, $\vec{a}$ decreases from 2 to 0 as the number of iterations. The change of $\vec{a}$ is shown in the formula

$$\vec{a} = 2 - \frac{2T}{T_{\max}}. \quad (16)$$

When choosing to swim to the position of a random whale, the swimming formula is shown in the following formula:

$$\vec{X}(t+1) = \vec{X}_{\text{rand}}(t) - \vec{A} \times \vec{D}, \quad (17)$$

where $\vec{X}_{\text{rand}}$ is the random position of the whale in the solution space. The value of the coefficient vector $\vec{A}$ determines how the whale swims during the enclosing phase. When $\vec{A} < 1$, move towards the optimal individual; when $\vec{A} > 1$, move towards a random individual.

4.1.2. Bubble Net Attack Prey. While whales use bubble nets to attack their prey, they also continuously update their position by directly constructing a spiral equation between themselves and their prey. The position update of the whale is shown in the following formulas:

$$\vec{D}' = \vec{X}_b(t) - \vec{X}(t), \quad (18)$$

$$\vec{X}(t+1) = \vec{D}' \times e^{bl} \times \cos(2\pi l) + \vec{X}_b(t). \quad (19)$$

In formula (19), b is a constant used to define the shape of a logarithmic spiral, and l is a random number uniformly distributed between $[-1, 1]$.

The task scheduling solution we proposed can be transformed into the whale position problem in the whale optimization algorithm [35]. The search process of whale swimming is the process of task finding a suitable virtual machine. The position of the leader whale is the optimal solution in the whole scheduling process, and the fitness function value of the leader is the minimum value of F_{obj} . In this way, we can use the WOA algorithm to schedule cloud computing tasks to get an optimal solution. In each iteration,

each whale must update its own position information according to the formula, put the position information into matrix A_{nm} for execution, calculate the fitness value of the whale at this position according to the fitness function F_{obj} , and find the optimal position is used as the position of the leader whale for the next iteration. Repeat this process until the iteration ends, and finally find the optimal solution of the entire population in matrix A_{nm} , that is, the best solution for performing scheduling.

4.2. Opposition-Based Learning. The concept of opposition-based learning (OBL) was proposed by Tizhoosh [47] in 2005. Its main idea is to generate reverse individuals based on existing individual information in the current space, so that reverse individuals participate in the competition with the current individual, compare the candidate solution and the reverse solution, and select a better individual to enter the iteration. Experiments prove that this mechanism is effective in improving the search efficiency of the algorithm.

Opposition-based learning is defined as follows: assuming that the solutions generated in D -dimensional space are $x_i = (x_{i1}, x_{i2}, \dots, x_{iD})$, $x_{ij} \in [lb, ub]$, lb and ub are the upper and lower limits of the solution space, respectively. Suppose its reverse solution is $x'_i = (x'_{i1}, x'_{i2}, \dots, x'_{iD})$, and the solution and reverse solution in the space satisfy the following formula:

$$x'_{ij} = k(lb_{ij} + ub_{ij}) - x_{ij}, \quad (20)$$

where k can be real numbers with different values. When $k = 0$, x'_i is a general OBL based on solving symmetry; when $k = 0.5$, x'_i is a general OBL based on interval symmetry. Here, we use generalized OBL, taking $k = 1$.

4.3. Gaussian Cloud Model. The cloud model [27] was proposed by Li Deyi et al. in 1995. It is an uncertainty conversion model between qualitative concepts and quantitative expressions expressed by linguistic values, which realizes the conversion between qualitative concepts and quantitative values. Cloud models have now been applied to various fields such as natural language processing and image processing. The cloud model has three numerical characteristics, which are expected E_x , entropy E_n , and super-entropy H_e , which are denoted as $C = (E_x, E_n, H_e)$. Expectation is the distribution expectation of cloud drops in space, which can represent a qualitative concept. Entropy is a measure of uncertainty in qualitative concepts, which can reflect randomness and ambiguity. The superentropy is the uncertainty measure of entropy, which is determined by the randomness and ambiguity of entropy.

The normal cloud model is an algorithm that basically obeys the normal distribution of cloud drops. After determining the expectations E_x and H_e , normal random numbers and cloud drops can be calculated to calculate the degree of membership. The calculation formula is as follows:

$$E'_{ni} = \text{Normrnd}(E_n, H_e), \quad (21)$$

$$x_i = \text{Normrnd}(E'_{ni}, |E'_{ni}|), \quad (22)$$

$$y_i = \exp\left[\frac{-(x_i - E_x)^2}{2(E'_{ni})^2}\right]. \quad (23)$$

The pseudocode of the Gaussian cloud model generation algorithm is shown in (Algorithm 1).

The distribution and digital characteristics of cloud drops in Gaussian clouds are shown in Figure 2.

Figure 2(a) represents a one-dimensional cloud model with expectation $E_x = -8$, entropy $E_n = 0.7$, superentropy $H_e = 0.1$, and cloud drop number $n = 5000$. Figure 2(b) represents a cloud model with expectation $E_x = 2.2$, entropy $E_n = 2$, superentropy $H_e = 0.1$, and cloud drop number $n = 5000$. Figure 2(c) represents a one-dimensional cloud model with $E_x = 18$, entropy $E_n = 4$, superentropy $H_e = 0.1$, and cloud drop number $n = 5000$. The digital characteristics of the cloud can be clearly seen through the three graphs in Figure 2.

5. Application of Whale Optimization Algorithm in Task Scheduling

The problem of searching for the best mapping between tasks and virtual machines in the process of task scheduling in cloud computing can be transformed into the problem of solving the optimal position of whales in the whale optimization algorithm. The search process of whale swimming is the process of task finding a suitable virtual machine. The position of the leader whale is the optimal solution in the entire scheduling process. The minimum value of the leader's fitness function value is the minimum value obtained by normalizing time, load, and cost in the entire scheduling process [34]. In this way, we can use the WOA algorithm to get an optimal solution in cloud computing task scheduling. In each iteration, each whale must update its own position information according to the formula, put the position information into the matrix A_{nm} for execution, and calculate the fitness value of the whale at this position according to the fitness function F_{obj} to find the optimal position is used as the position of the leader whale for the next iteration. Repeat this process until the iteration ends, and finally find the optimal solution of the entire population in the matrix A_{nm} , that is, the best solution to perform scheduling.

6. Multiobjective Task Scheduling Strategy Based on Gaussian Whale-Cloud Optimization in Cloud Computing (GCWOAS2)

Our research found that the whale optimization algorithm randomly generates an initial population at the beginning of the iteration, which cannot guarantee the diversity of the population, and will affect the search efficiency of the algorithm to a certain extent. At the same time, the algorithm does not jump out of the local optimal solution, and it is easy

to fall into the local optimal when it converges quickly. Therefore, by making the following improvements to the whale optimization algorithm, we propose a multiobjective task scheduling strategy GCWOAS2 based on Gaussian cloud-whale optimization in cloud computing: (1) generate its reverse population while randomly generating the population to increase the diversity of the population. (2) In the search process, an adaptive movement factor is introduced to expand the search range, so that the algorithm jumps out of the local optimum. (3) The normal cloud model is introduced to make the algorithm enhance the global search and jump out of the local optimum.

6.1. Initial Population Algorithm Based on Opposition-Based Learning Mechanism. Since the randomly generated population has not been estimated, it has greater uncertainty, and the convergence rate is unpredictable, so we consider the reverse individual of each individual at the same time, and choose the minimum fitness function value shown in formula (10) as the initial individual, each individual in the population will be closer to the optimal solution, so in the initialization phase, we introduced a reverse learning mechanism to initialize the population.

The pseudocode of the initial population algorithm based on opposition-based learning mechanism is shown in Algorithm 2.

6.2. The Formula of Dynamically Adjusting the Moving Step Length Based on the Adaptive Moving Factor. As shown in the above content, formula (13) is mainly used to calculate the encircling step size $\vec{A} \times \vec{C}$ of the WOA algorithm, and the encircling step size can affect the search range of the algorithm. When placed in the task scheduling of cloud computing, it can intuitively affect the range of tasks to search the suitable VM. It can be seen from formula (14) that the change of $\vec{C}$ is directly related to the change of $\vec{r}_2$, so we propose to improve $\vec{r}_2$ according to formula (21) as follows:

$$\vec{r}_2 = \tan\left(\frac{\pi}{4} \times \frac{t}{\text{Maxiter}}\right), \quad (24)$$

where t represents the current iteration number of the population, and Maxiter represents the maximum iteration number of the population. We can see that $\vec{r}_2$ gradually becomes larger as the number of iterations increases, and $\vec{A}$ also gradually becomes larger, and the moving step size increases accordingly. In the initial stage of the search, we can find that the step size is small, which can effectively limit the global search in the population and maintain the diversity of the population; in the later stage of the search, the step size starts to increase gradually, and the global search ability is enhanced, which is convenient for the algorithm to jump out of the local area. So that the global optimal solution can be found.

In addition, we also used the Gaussian cloud model to improve the whale's swimming process to further improve the algorithm's search ability in cloud task scheduling.

Input: the expectations E_x , entropy H_e , and cloud drop number N

Output: membership function value y_i

- (1) **For** $i = 1$ **to** N **do**
- (2) Generate a normal distribution random number E'_{mi} according to formula (21)
- (3) Generate a normal distribution random number x_i according to formula (22)
- (4) Calculate the membership function y_i by formula (23)
- End for

ALGORITHM 1: Gaussian cloud model generation.

FIGURE 2: Gaussian cloud model under different conditions. (a) $E_x = -8, E_n = 0.7, H_e = 0.1$, and $n = 5000$. (b) $E_x = 2.2, E_n = 2, H_e = 0.1$, and $n = 5000$. (c) $E_x = 18, E_n = 4, H_e = 0.1$, and $n = 5000$.

6.3. Whale Optimization Algorithm Based on Gaussian Cloud Model. The Gaussian cloud model is introduced into the whale optimization algorithm to improve the walking step length of the whale and optimize the random swimming around the prey during the swimming search and the process of driving the prey by the bubble net. Due to the randomness of the Gaussian cloud model, tasks can be matched to more different types of VM in the cloud computing task scheduling process, which expands the search scope, improves the probability of matching to different

VMs, and improves the convergence speed. The global optimization capability has also been effectively improved.

6.3.1. Random Walk Process When Surrounding Prey. In order to make the algorithm expand the search range out of the local optimum during the search process, we use the randomness of the Gaussian cloud model to update the position in the random walk process, and the process is as follows:

Input: the number of search agent Num , upper bound of solution space lb , and lower bound ub
Output: new population PB

- (1) **For** $i = 1$ **to** Num **do**
- (2) Randomly generate an initial population P
- (3) According to formula (20), the opposite population PW of population P is generated
- (4) Calculate the fitness of P and PW according to the fitness function (11) in order
- (5) Select individuals with higher fitness and put them in the final population PB
- (6) **End for**

ALGORITHM 2: Population initialization based on OBL.

$$E_x = X_i, \quad (25)$$

$$E_n = \frac{|X_b - X_w|}{3}, \quad (26)$$

$$H_e = \frac{E_n}{r_2}, \quad (27)$$

$$E'_n = N \cdot (E_n, H_e), \quad (28)$$

$$\mu = e^{-((X_i - E_x)^2 / 2 (E'_n)^2)}, \quad (29)$$

where X_i represents the current position, E_x represents the expectation of the Gaussian cloud model, X_b and X_w represent the most and worst positions of the whale, respectively, E_n is the entropy in the Gaussian cloud model, H_e represents the superficial value in the Gaussian cloud model, E'_n represents a random value between entropy and super-entropy, and μ represents the value of the membership function.

The new position obtained after random walk is shown in the following formula:

$$\vec{X}(t+1) = \mu \times (\vec{X}_{\text{rand}}(t) - \vec{A} \times \vec{D}). \quad (30)$$

6.3.2. Bubble Net to Drive Away Prey. Here, we propose a method of using the randomness of the Gaussian cloud model to improve the spiral update method of the bubble net in the process of driving prey, increase the choice of the spiral path of the whale in the optimization process, and make the whale have a certain degree in the search process. The randomness increases the search range of the whale group and improves the global search ability of the algorithm:

$$E_x = \frac{(ub - lb)}{20} \times \left(\frac{1 - t}{\text{Iters}} \right)^2, \quad (31)$$

$$E_n = \text{abs} \frac{(X_b - X_w)}{3}, \quad (32)$$

$$H_e = \frac{E_n}{3}, \quad (33)$$

$$E'_n = \text{normrnd}(E_n, H_e), \quad (34)$$

$$w = \text{rand}(1) \times E'_n, \quad (35)$$

$$S = \exp \frac{-(w - E_n)^2}{(2 \times E_n'^2)}. \quad (36)$$

In this step, the parameters in the formula have the same meaning as in the previous text, and S represents the value of the membership function in the Gaussian cloud model.

Then, the bubble net drives the prey to update the individual position as shown in

$$\vec{X}(t+1) = S \times \left(\vec{D}^i \times e^{Sbl} \times \cos(2\pi l) + S \times \vec{X}_b(t) \right). \quad (37)$$

According to the basic Gaussian cloud model shown in Algorithm 1, the Gaussian cloud model is introduced into the whale swimming process, and a whale optimization algorithm based on the Gaussian cloud model is proposed, as shown in Algorithm 3.

6.4. Multiobjective Task Scheduling Algorithm-GCWOA Algorithm Based on Gaussian Whale-Cloud Optimization in Cloud Computing. Based on the above design, we propose a multiobjective task scheduling algorithm based on Gaussian whale-cloud optimization (GCWOA) for cloud computing task scheduling, and the flowchart is shown in Figure 3. Its main realization can be divided into the following processes:

- (i) Step 1: this step is mainly used to design the mapping relationship between tasks and whales in cloud computing, and the initial position of each whale. In addition, we also initialize the dimensions of the search space, the upper and lower bounds of the search space, and the maximum number of iterations of the population.
- (ii) Step 2: introduce a reverse learning mechanism to generate a suitable scheduling plan in the process of generating the initial population, and expand the search range.
- (iii) Step 3: after the initialization is completed, the algorithm begins to iterate. In this step, the optimal

Input: the upper bound lb and lower bound ub of the solution space, the number of iterations $iters$, and number of search agent Num

Output: The updated location of the whale $\vec{X}(t+1)$.

- (1) **For** $t = 1$ **to** $iters$ **do**
- (2) **For** $i = 1$ **to** Num **do**
- (3) Record current individual X_i , select the best individual in the population X_b and the worst individual X_w .
- (4) **if** $p < 0.5$
- (5) Calculate the membership function μ of the Gaussian cloud model according to formula (29)
- (6) The membership function is used to update the new position of the whale in the swimming process as shown in formula (30).
- (7) **else**
- (8) According to the upper and lower bounds of the solution space, the number of iterations is dynamically adjusted E_x , as in formula (31).
- (9) Calculate the membership function S of the Gaussian cloud model according to formula (36).
- (10) The membership function is used to update the new position of the whale in the swimming process as shown in formula (37).
- (11) **End for**
- (12) **End for**

ALGORITHM 3: Whale optimization algorithm based on the Gaussian cloud model.

FIGURE 3: GCWOA algorithm flowchart.

position and the worst position of the whale are calculated according to the fitness function we proposed to calculate the time, load, and cost (refer to formula (10)), that is, the best scheduling plan and the worst plan in the scheduling process. At the

same time, the coefficient vector is calculated according to the adaptive adjustment factor.

- (iv) Step 4: determine the swimming mode of the whale according to the vector $\vec{A}$, and guide the whale to randomly swim to find the optimal position

according to the process in Algorithm 3, or spiral search to update the position.

- (v) Step 5: when all the whale positions are updated, that is, all scheduling plans are found, the next iteration will be performed. When the maximum number of iterations is reached, the algorithm terminates, otherwise go to Step 2 to continue to update the position of the whale. After the iteration is completed, the optimal position, that is, the most suitable task and virtual machine mapping, will be transferred to matrix A to obtain the best task scheduling plan.

According to the above steps, the complete GCWOA algorithm we get is shown in Algorithm 4, and the algorithm flow is shown in GCWOA.

7. Simulation Results and Analysis

In this part, we performed simulation verification and performance evaluation on the GCWOA algorithm proposed in this article and compared it with other existing algorithms. In this experiment, we used CPU (AMD quad-core, 2.4 GHz), memory (4.0 GB), Windows 10 operating system, and MatlabR2016b as the simulation platform and carried out the proposed GCWOA algorithm with common metaheuristic algorithms such as ACO and PSO. For comparison, the algorithm is compared with the original WOA algorithm described in the article. According to our simulation results, we compare and evaluate the performance of these algorithms in cloud computing task scheduling.

7.1. Experimental Data Settings and Indicators. The performance evaluation indicators of our algorithm include time, virtual machine load, virtual machine cost, and the total cost of the final model. Our experiment mainly evaluates the performance of each algorithm in a small number of tasks. The main parameters used by each algorithm in the experiment are shown in Table 1, and the VM and task parameters are shown in Table 2. In addition, we set the number of VMs to 40, and the number of search agents for all algorithms is set to 50 (that is, there are 50 sets of solutions). As the number of iterations increases, the algorithm will get closer and closer to the optimal solution, which is of great significance to cloud computing task scheduling [48], so we set the number of iterations to 100.

7.2. Experimental Results and Analysis. We implement and evaluate the performance of each algorithm by increasing the number of iterations and changing the number of tasks. The experimental result table is shown in Table 3. From Table 3, we can see the value of the fitness function of each algorithm when iterating 100 times and the number of tasks is 100 and 1000.

In Figure 4, we can see that as the number of iterations increases, the GCWOA algorithm is better than other algorithms in terms of total cost, which effectively proves the role of our algorithm in the task scheduling process. Since the opposition-based learning mechanism is introduced when the GCWOA algorithm initializes the population, the GCWOA algorithm is closer to the optimal value than other algorithms at the beginning of the iteration. The existence of the adaptive adjustment factor makes the algorithm search a larger range in the search process, so as to find more different solutions than other algorithms. The introduction of the Gaussian cloud model enables the GCWOA algorithm to jump out of the local optimal and find the global optimal solution. In addition, we can see that the optimization effect of the GCWOA algorithm we proposed is the best. Compared with the three algorithms of WOA, PSO, and ACO, the total cost of the GCWOA algorithm is reduced. At the same time, we can see that the GCWOA algorithm converges rapidly at the beginning of the iteration, and has stabilized at the 16th iteration, indicating that the GCWOA algorithm can converge to the optimal solution relatively quickly.

The time cost and virtual machine load are shown in Figures 4(b) and 4(c). In these two figures, the values of the four algorithms are decreasing with the increase of the number of iterations, indicating that the four algorithms are searching in the process, and costs can be effectively reduced. It is worth mentioning that the GCWOA algorithm has much better performance than the other three algorithms in terms of convergence accuracy.

However, compared with Figures 4(a)–4(c) in Figure 4, the virtual machine cost shown in Figure 4(d) is different. Although the value of the four algorithms in processing virtual machine costs fluctuates with the increase in the number of iterations, from a global perspective, the PSO, WOA, and GCWOA algorithms all decrease significantly and then increase significantly until they reach a stable state.

This shows that these three algorithms consume a large amount of virtual machine resources at a certain stage. In general, the GCWOA algorithm does not optimize this cost well. The reason for this problem may be due to multi-objective scheduling. In order to achieve the global optimal solution, the GCWOA algorithm sacrifices a certain local optimal value (here, the virtual machine cost) in the search process to balance the two. At the same time, it also proves that the GCWOA algorithm can be used to solve complex scheduling problems.

We also evaluated the performance of the four algorithms with the number of iterations under large-scale tasks. As shown in Figure 5(a), although the total cost of the four algorithms has increased compared with the amount of 100 tasks, the total cost of each algorithm decreases as the number of iterations increases, which shows the effectiveness of the algorithm in cloud computing task scheduling. In addition, the algorithms converge quickly at the beginning of the iteration. The GCWOA algorithm continues to converge until it stabilizes at the 20th iteration, indicating

Input: task parameter N , VM parameter M , number of iterations $Iters$, number of search agent Num

Output: the best F_{obj} and matrix A (Task-VM mapping).

```

(1) Begin
(2)   Generate a population NOP according to Algorithm 1.
(3)   For  $t = 1$  to  $Iters$  do
(4)     For  $i = 1$  to  $Num$  do
(5)       Calculate the optimal solution  $g_b$  and worst solution  $g_w$  of each scheme according to the fitness function of formula (24).
(6)       The coefficient vectors  $\vec{A}$  and  $\vec{C}$  are calculated, and the probability  $P$  is randomly generated.
(7)       if ( $p < 0.5$ ) do
(8)         if ( $|\vec{A}| \geq 1$ ) do
(9)           Calculate the membership function  $\mu$  of the Gaussian cloud model according to formula (28).
(10)          The membership function is used to update the new position of the whale in the swimming process as shown in
              formula (30).
(11)         else
(12)           Use formula (13) to update the whale position.
(13)         End if
(14)       else
(15)         According to the upper and lower bounds of the solution space, the number of iterations is dynamically adjusted  $E_x$ , as
              in formula (31).
(16)         According to formula (36), the membership function  $S$  of the Gaussian cloud model is calculated.
(17)         The membership function is used to update the new position of the whale in the swimming process as shown in formula
              (37).
(18)       End if
(19)     End For
(20)   End For
Output: the best  $F_{obj}$  and matrix  $A$  (Task-VM mapping).

```

ALGORITHM 4: Multiobjective task scheduling algorithm based on the Gaussian whale-cloud optimization-GCWOA algorithm.

TABLE 1: The main parameters of the algorithm.

Algorithm	Parameter	Value	Description
ACO	ρ	0.7	Pheromones
	p	0.3	Probability of path selection
PSO	ω	0.9	Inertia weight
	c_1	1.8	Acceleration constant
	c_2	1.8	Acceleration constant
GCWOA(WOA)	b	1	Spiral search path parameters
	lb	-5	Upper bound of solution space
	ub	10	Lower bound of solution space

TABLE 2: VM and task parameters.

Parameter	Value range
CPU (VM)	[200, 500]
Memory (VM)	[100, 500]
Bandwidth (VM)	[100, 250]
Length (Task)	[50, 100]

that the algorithm has jumped out of the local optimum and found the most suitable scheduling scheme. It also proves that the strategy we proposed is effective.

Compared with the other three algorithms, the total cost of our proposed GCWOA algorithm is significantly lower than other algorithms, which proves that the GCWOA algorithm can also optimize large-scale task scheduling. The time cost and load cost of the four algorithms are shown in

TABLE 3: Experimental results after 100 iterations.

Number of tasks	Target value	Algorithm			
		ACO	PSO	WOA	GCWOA
Task100	F1	0.3012	0.2698	0.2690	0.2623
	F2	0.1948	0.1685	0.1533	0.1489
	F3	0.3833	0.3424	0.3370	0.3302
	F4	0.3255	0.2986	0.3168	0.3098
Task1000	F1	0.5954	0.5322	0.5576	0.5245
	F2	0.3779	0.3701	0.3749	0.3684
	F3	0.6569	0.4659	0.5362	0.4440
	F4	0.7515	0.7606	0.7618	0.7599

Figures 5(b) and 5(c). We can see that the GCWOA algorithm can more effectively use resources when processing large-scale tasks and reduce task execution time. At the same time, it

FIGURE 4: The best performance of 100 tasks with the number of iterations. (a) The value of total cost. (b) The value of time cost. (c) The value of load cost. (d) The value of price cost.

reduces resource consumption, saves memory, and reduces the underlying load cost. In the iterative process, the algorithm converges rapidly. After reaching the local optimum, the curve continues to decrease until it is stable, which reflects the advantages of the GCWOA algorithm in the scheduling of fast convergence and easy to jump out of the local optimum.

As shown in Figure 5(d), the virtual machine operating cost is different when processing large-scale tasks and small-scale tasks. Although the cost of GCWOA is still higher than other algorithms, when we consider the total cost, the GCWOA algorithm is still superior to other algorithms and has a stronger global search capability than other algorithms.

FIGURE 5: The performance of 1000 tasks with the number of iterations. (a) The value of total cost. (b) The value of time cost. (c) The value of load cost. (d) The value of price cost.

8. Conclusion

The rapid increase in the number of user tasks has led to data flooding, low scheduling efficiency, and unbalanced resource allocation, making task scheduling in cloud computing become an agenda. This paper proposes a “whale-Gaussian cloud” model, which combines heuristic algorithms and Gaussian cloud to achieve better scheduling. On the basis of this model, we propose a scheduling algorithm based on GCWOA. In addition to combining the whale optimization algorithm with the Gaussian cloud model, we also introduce an opposite learning mechanism and an adaptive adjustment factor into the algorithm. The opposite learning mechanism is used to generate more suitable task-VM mapping. Both the adaptive adjustment factor and the Gaussian cloud

model can guide the algorithm in the whale optimization algorithm to select the optimal scheduling scheme in cloud computing. We have conducted a large number of experiments to verify the performance of our proposed scheduling strategy. The experimental results show that the algorithm proposed in this paper has ideal effects in processing small tasks and large tasks and can reduce task completion time and balance load of virtual machines.

However, no matter how large the task is, the algorithm in this paper does not perform well in terms of operating costs. How to make the GCWOAS2 strategy further reduce operating costs will be the focus of our future research. In the future, we will continue to study the application of heuristic algorithms in multiobjective scheduling to enable more effective and reasonable scheduling of resources in cloud computing nodes.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Training Program of the Major Research Plan of NSF of China under grant 91746104, Humanity and Social Science Fund of the Ministry of Education under grant 20YJAZH078 and 20YJAZH127, Open Project of Tongji University Embedded System and Service Computing of Ministry of Education of China under grant ESSCKF 2019-06 and ESSCKF 2019-08, and Science Education and Industry Integration Innovation Pilot Project of Qilu University of Technology (Shandong Academy of Sciences) under grant 2020KJC-ZD03.

References

- [1] A. Weiss, "Computing in the clouds," *Networker*, vol. 11, no. 4, pp. 16–25, 2007.
- [2] A. N. Toosi, R. N. Calheiros, and R. Buyya, "Interconnected cloud computing environments," *ACM Computing Surveys*, vol. 47, no. 1, pp. 1–47, 2014.
- [3] M. Cusumano, "Cloud computing and SaaS as new computing platforms," *Communications of the ACM*, vol. 53, no. 4, pp. 27–29, 2010.
- [4] S. K. Gavvala, C. Jathoth, G. R. Gangadharan, and R. Buyya, "QoS-aware cloud service composition using eagle strategy," *Future Generation Computer Systems*, vol. 90, pp. 273–290, 2019.
- [5] L. Ni, J. Zhang, C. Jiang, C. Yan, and K. Yu, "Resource allocation strategy in fog computing based on priced timed petri nets," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1216–1228, 2017.
- [6] Z. Lv and W. Xiu, "Interaction of edge-cloud computing based on SDN and NFV for next generation IoT," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 5706–5712, 2020.
- [7] S. Singh and I. Chana, "A survey on resource scheduling in cloud computing: issues and challenges," *Journal of Grid Computing*, vol. 14, no. 2, pp. 217–264, 2016.
- [8] S. K. Garg, A. N. Toosi, S. K. Gopalaiyengar, and R. Buyya, "SLA-based virtual machine management for heterogeneous workloads in a cloud datacenter," *Journal of Network and Computer Applications*, vol. 45, pp. 108–120, 2014.
- [9] X. Zuo, G. Zhang, and W. Tan, "Self-adaptive learning PSO-based deadline constrained task scheduling for hybrid IaaS cloud," *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 2, pp. 564–573, 2014.
- [10] J.-W. Lin, C.-H. Chen, and C.-Y. Lin, "Integrating QoS awareness with virtualization in cloud computing systems for delay-sensitive applications," *Future Generation Computer Systems*, vol. 37, pp. 478–487, 2014.
- [11] C. Delimitrou and C. Kozyrakis, "QoS-aware scheduling in heterogeneous datacenters with paragon," *ACM Transactions on Computer Systems*, vol. 31, no. 4, pp. 1–34, 2013.
- [12] S. T. Milan, L. Rajabion, A. Darwesh, M. Hosseinzadeh, and N. J. Navimipour, "Priority-based task scheduling method over cloudlet using a swarm intelligence algorithm," *Cluster Computing*, vol. 23, no. 2, pp. 663–671, 2020.
- [13] I. Gupta, M. S. Kumar, and P. K. Jana, "Efficient workflow scheduling algorithm for cloud computing system: a dynamic priority-based approach," *Arabian Journal for Science and Engineering*, vol. 43, no. 12, pp. 7945–7960, 2018.
- [14] B. Keshanchi and N. J. Navimipour, "Priority-based task scheduling in the cloud systems using a memetic algorithm," *Journal of Circuits, Systems and Computers*, vol. 25, no. 10, Article ID 1650119, 2016.
- [15] P. Akilandeswari, B. Vennila, and H. Srimathi, "Cluster based scheduling method with task duplication in cloud platform," *AIP Conference Proceedings*, vol. 2112, Article ID 020051, 2019.
- [16] M. H. Moghadam and S. M. Babamir, "Makespan reduction for dynamic workloads in cluster-based data grids using reinforcement-learning based scheduling," *Journal of Computational Science*, vol. 24, pp. 402–412, 2018.
- [17] T. Springer and E. Linstead, "Adaptive QoS-based resource management framework for IoT/edge computing," in *Proceedings of the 2018 9th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pp. 210–217, New York, NY, USA, November 2018.
- [18] S. Xue, Y. Zhang, X. Xu et al., "QET: a QoS-based energy-aware task scheduling method in cloud environment," *Cluster Computing*, vol. 20, no. 4, pp. 3199–3212, 2017.
- [19] A. Arunarani, D. Manjula, and V. Sugumaran, "Task scheduling techniques in cloud computing: a literature survey," *Future Generation Computer Systems*, vol. 91, pp. 407–415, 2019.
- [20] X. Wei, "Task scheduling optimization strategy using improved ant colony optimization algorithm in cloud computing," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–12, 2020.
- [21] G. Sun, C. Li, and L. Deng, "An adaptive regeneration framework based on search space adjustment for differential evolution," *Neural Computing and Applications*, 2021.
- [22] F. Pop, C. Dobrescu, V. Cristea, N. Bessis, F. Xhafa, and L. Barolli, "Reputation-guided evolutionary scheduling algorithm for independent tasks in inter-clouds environments," *International Journal of Web and Grid Services*, vol. 11, no. 1, pp. 4–20, 2015.
- [23] M. A. Tawfeek, A. El-Sisi, A. E. Keshk, and F. A. Torkey, "Cloud task scheduling based on ant colony optimization," in *Proceedings of the 2013 8th International Conference on Computer Engineering & Systems (ICCES)*, pp. 64–69, Cairo, Egypt, November 2013.
- [24] J. Li, J. Zhang, C. Jiang, and M. Zhou, "Composite particle swarm optimizer with historical memory for function optimization," *IEEE Transactions on Cybernetics*, vol. 45, no. 10, pp. 2350–2363, 2015.
- [25] Q. Kang, M. Zhou, J. An, and Q. Wu, "Swarm intelligence approaches to optimal power flow problem with distributed generator failures in power networks," *IEEE Transactions on Automation Science and Engineering*, vol. 10, no. 2, pp. 343–353, 2013.
- [26] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, pp. 51–67, 2016.
- [27] M. Wang, Y. Wang, F. Shen, and J. Jin, "A novel classification approach based on integrated connection cloud model and game theory," *Communications in Nonlinear Science and Numerical Simulation*, vol. 93, Article ID 105540, 2021.
- [28] C.-W. Tsai and J. J. P. C. Rodrigues, "Metaheuristic scheduling for cloud: a survey," *IEEE Systems Journal*, vol. 8, no. 1, pp. 279–291, 2014.

- [29] Y. Xiong, S. Huang, M. Wu, J. She, and K. Jiang, "A Johnson's-rule-based genetic algorithm for two-stage-task scheduling problem in data-centers of cloud computing," *IEEE Transactions on Cloud Computing*, vol. 7, no. 3, pp. 597–610, 2019.
- [30] S. Wang, A. Zhou, C.-H. Hsu, X. Xiao, and F. Yang, "Provision of data-intensive services through energy- and QoS-aware virtual machine placement in national cloud data centers," *IEEE Transactions on Emerging Topics in Computing*, vol. 4, no. 2, pp. 290–300, 2016.
- [31] J. Zhao, K. Yang, X. Wei, Y. Ding, L. Hu, and G. Xu, "A heuristic clustering-based task deployment approach for load balancing using Bayes theorem in cloud environment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 2, pp. 305–316, 2016.
- [32] D. Ding, X. Fan, Y. Zhao, K. Kang, Q. Yin, and J. Zeng, "Q-learning based dynamic task scheduling for energy-efficient cloud computing," *Future Generation Computer Systems*, vol. 108, pp. 361–371, 2020.
- [33] M. S. Sanaj and P. M. Joe Prathap, "An efficient approach to the map-reduce framework and genetic algorithm based whale optimization algorithm for task scheduling in cloud computing environment," *Materials Today: Proceedings*, vol. 37, no. 1, 2020.
- [34] L. D. Dhinesh Babu and P. Venkata Krishna, "Honey bee behavior inspired load balancing of tasks in cloud computing environments," *Applied Soft Computing*, vol. 13, no. 5, pp. 2292–2303, 2013.
- [35] X. Chen, L. Cheng, C. Liu et al., "A WOA-based optimization approach for task scheduling in cloud computing systems," *IEEE Systems Journal*, vol. 14, no. 3, pp. 3117–3128, 2020.
- [36] L. Zuo, L. Shu, S. Dong, C. Zhu, and T. Hara, "A multi-objective optimization scheduling method based on the ant colony algorithm in cloud computing," *IEEE Access*, vol. 3, pp. 2687–2699, 2015.
- [37] M. Hosseinzadeh, M. Y. Ghafour, H. K. Hama, B. Vo, and A. Khoshnevis, "Multi-objective task and workflow scheduling approaches in cloud computing: a comprehensive review," *Journal of Grid Computing*, vol. 18, no. 3, pp. 327–356, 2020.
- [38] A. Mohammadzadeh, M. Masdari, F. S. Gharehchopogh, and A. Jafarian, "A hybrid multi-objective metaheuristic optimization algorithm for scientific workflow scheduling," *Cluster Computing*, vol. 24, no. 2, pp. 1–25, 2020.
- [39] J. Xu, Z. Zhang, Z. Hu, L. Du, and X. Cai, "A many-objective optimized task allocation scheduling model in cloud computing," *Applied Intelligence*, vol. 51, 2020.
- [40] Z. Peng, J. Lin, D. Cui, Q. Li, and J. He, "A multi-objective trade-off framework for cloud resource scheduling based on the deep Q-network algorithm," *Cluster Computing*, vol. 23, no. 4, pp. 2753–2767, 2020.
- [41] C.-L. Huang, Y.-Z. Jiang, Y. Yin, W.-C. Yeh, V. Y. Y. Chung, and C.-M. Lai, "Multi objective scheduling in cloud computing using MOSSO," in *Proceedings of the 2018 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8, Rio de Janeiro, Brazil, July 2018.
- [42] H. Chen, G. Wu, W. Pedrycz, P. N. Suganthan, L. Xing, and X. Zhu, "An adaptive resource allocation strategy for objective space partition-based multiobjective optimization," *IEEE Transactions on System, Man, and Cybernetics: Systems*, vol. 51, no. 3, pp. 1507–1522, 2021.
- [43] L. Wang and X.-l. Zheng, "A knowledge-guided multi-objective fruit fly optimization algorithm for the multi-skill resource constrained project scheduling problem," *Swarm and Evolutionary Computation*, vol. 38, pp. 54–63, 2018.
- [44] I. M. Ali, K. M. Sallam, N. Moustafa, R. Chakraborty, M. J. Ryan, and K.-K. R. Choo, "An automated task scheduling model using non-dominated sorting genetic algorithm II for fog-cloud systems," *IEEE Transaction on Cloud Computing*, 2020.
- [45] K. R. Kumari, P. Sengottuvelan, and J. Shanthini, "A hybrid approach of genetic algorithm and multi objective PSO task scheduling in cloud computing," *Asian Journal of Research in Social Sciences and Humanities*, vol. 7, no. 3, p. 1260, 2017.
- [46] M. Armbrust, A. Fox, R. Griffith et al., "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.
- [47] H. R. Tizhoosh and S. Mehta, "Pattern Analysis and Machine Intelligence Laboratory, Systems Design Engineering, "Opposition-Based Reinforcement Learning," *Journal of Advanced Computational Intelligence and Intelligent Informatics*, vol. 10, pp. 578–585, 2006.
- [48] P. Kaur and S. Mehta, "Resource provisioning and work flow scheduling in clouds using augmented shuffled frog leaping algorithm," *Journal of Parallel and Distributed Computing*, vol. 101, pp. 41–50, 2017.