QoS-aware and Multi-Objective Virtual Machine Dynamic Scheduling for Big Data Centers in Clouds

Jirui Li (✉ ljrokyes@163.com)

Henan University of Chinese Medicine

Rui Zhang

Arizona State University

Yafeng Zheng

Henan University of Economics and Law

Research Article

Keywords: Load Balancing , Migration Cost , QoS , Big Data Center , Scheduling , Cloud

Posted Date: November 3rd, 2021

DOI: <https://doi.org/10.21203/rs.3.rs-986915/v1>

License: This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

QoS-aware and Multi-Objective Virtual Machine Dynamic Scheduling for Big Data Centers in Clouds

Jirui Li · Rui Zhang · Yafeng Zheng*

Received: date / Accepted: date

Abstract Efficient resource scheduling is one of the most critical issues for big datacenters in clouds to provide continuous services for users. Many existing scheduling schemes based on tasks on virtual machine (VM), pursued either load balancing or migration cost under certain response time or energy efficiency, which cannot meet the true balance of the supply and demand between users and cloud providers. The paper focuses on the following multi-objective optimization problem: how to pay little migration cost as much as possible to keep system load balancing under meeting certain quality of service (QoS) via dynamic VM scheduling between limited physical nodes in a heterogeneous cloud cluster. To make these conflicting objectives coexist, a joint optimization function is designed for an overall evaluation on the basis of a load balancing estimation method, a migration cost estimation method and a QoS estimation method. To optimize the consolidation score, an array mapping and a tree crossover model are introduced, and an improved genetic algorithm (GA) based on them is proposed. Finally, empirical results based on Eucalyptus platform demonstrate the proposed scheme outperforms exiting VM scheduling models.

Keywords Load Balancing · Migration Cost · QoS · Big Data Center · Scheduling · Cloud

Jirui Li
School of Information Technology, Henan University of Chinese Medicine, Zhengzhou 450008, China
E-mail: ljrokyes@163.com

Rui Zhang
School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Tempe, AZ 85281 USA

Yafeng Zheng (*Corresponding Author)
School of Computer and Information Engineering, Henan University of Economics and Law, Zhengzhou 450008, China

1 INTRODUCTION

Currently, billions of heterogeneous physical devices are collected together by the Internet and forms a variety of intelligent applications such as smart e-healthcare, smart home, intelligent transportation and smart city, which not only brings great convenience to human life, but also produces massive data that need big storage and high computational capabilities to deal with [1]. Cloud computing is popular for scalable computation and processing of big data to overcome the limitations of individual devices [2]. However, with the rapid increase of networked intelligent devices, it is becoming more compelling for the imbalance workload problem of data centers at any time caused by the uncontrollability of the location and quantity of access devices.

The virtualization, that refers to the logical creation of an Internet resource that does not physically exist, is still a common resource balancing technology in heterogeneous data centers. Many virtualization based studies, such as [2–21], had developed various of schedule solutions with different goals to dynamically adjust the workload of data center infrastructures, reduce the cost of network and improve QoS of cloud. For example, [3] explored the task scheduling algorithm using a hybrid approach, which combined the genetic algorithms and the bacterial foraging algorithms, to minimize the makespan and second and reduce the energy consumption in the cloud computing. A hybrid discrete artificial bee colony algorithm was proposed in [4] to solve the flexible task scheduling problem in a cloud computing system. It mainly focused on the minimization of the maximum completion time and the maximization of total workloads of all devices. [5] achieved geographical load balancing via routing workloads dynamically to reduce overall energy consumption. [6] leveraged a

modified Particle Swarm Optimization algorithm to locate the tasks for VMs and then adopted a HYBRID Bio-Inspired algorithm to handle the allocation of resources (CPU and Memory). It only considered reducing the average response time. [7] developed a scalable algorithm to find a redirection of tasks between a given set of edge data centers in a network, thus minimizing the maximum of average response time.

Although these studies can improve the efficiency of cloud services to some extent, they have some key limitations: *First*, migration cost, response time, energy efficiency, QoS, security and privacy can be regarded as the targets to realize cloud load balancing. However, the above literatures nearly focused on one or two of these goals, which would cause the experimental results to deviate from the actual needs and have certain one sidedness. *Second*, physical resources are mainly consumed by tasks running on VMs, and nearly all studies leveraged the method of task scheduling to realize system load balancing based on the analysis of task traffic. Obviously, they did not consider the limited physical resources and assumed that the virtual pools were enough to run the assigned tasks. *Besides*, in distributed environments, there are two main load balancing methods: static and dynamic. The main disadvantage of the former is that it does not consider the state of the target VM or physical node when judging whether to migrate or not. However, although the latter is very difficult to achieve, it can always provide better solutions for the realization of real-time and sustainable cloud services.

Obviously, it provides a chance for us to improve the utilization of cloud resources due to the limitations of the above conditions. This paper proposes a QoS-aware and multi-objective VM dynamic scheduling model for the assignment of heterogeneous cloud resources, and its main contributions are as follows.

- *The proposed model focuses on the scheduling of VMs, which solves dynamic allocation problem of VMs on the limited physical infrastructures in the big data cloud centers.* At current, although the physical infrastructures have many advantages such as a cheap price and a low cost to add at anytime and anywhere, their capabilities are always limited to quickly and real-time response the requests from the vast user base in the actual network applications. It is a reliable solution via leveraging a realistic plan with high performance to allocate VM resources on them dynamically.
- *Leveraging three objects including QoS, migration cost and load balancing and an improved genetic algorithm to solve the mapping between VMs and physical nodes, and then developing the dynamic scheduling function of our proposed model.* High QoS, low

migration cost and the balance of resource utilization are the common indexes measuring the scheduling performance. Differing from previous studies that might only use one or two of them, our proposed model simultaneously uses these three indicators to build the objective functions, which can provide more reliable guarantee for users. In addition, our model dynamically adjusts the resource balance according to the running state of VMs on physical nodes, whereas the previous dynamic scheduling models are based on data traffic mainly.

- *A system is built to simulate a big data center in cloud, and our proposed model is evaluated via comparing with Greedy [23], Round Robin [24] and the Non-dominated Sorting Genetic Algorithm (NSGA II) [25] in Eucalyptus.* The experimental results demonstrate that our model significantly outperforms the representative Round Robin, Greedy and NSGA II, especially in dynamic and unbalanced data flow distribution.

The rest of the paper is organized as follows. In Section II, we introduce the previous related work. Section III provides the framework of our proposed model and the corresponding problem statement. In Section IV, we described the detailed calculations of the proposed algorithm. Section V builds the experimental settings and analyzes the performance of our model. Section VI concludes this paper.

2 RELATED WORK

Scheduling VMs is the common method for the resource scheduling when some system resources have no extra capacity to handle the next tasks. To improve the users' experience, many efforts have done to minimize the response time and processing time, load balancing, migration cost and meet the service level agreement (SLA) and so on.

To realize the load balancing of backbone network, many efficient VM scheduling schemes such as [8–14] have been proposed. [8] solved VM scheduling for load balancing via using a weighted Round Robin algorithm in Eucalyptus platform, and its advantage was that it could allocate all incoming requests to the available VMs in Round Robin fashion based on the weights without considering the current load on each VM. However, it did not validate the rationality of the weighted method and handled the users' requests too simple. Similarly, [9] and [10] proposed the adaptive algorithm for dynamic priority based VM scheduling in cloud to overcome the drawback of the existing algorithms in Eucalyptus. They could prevent a particular node from

being overloaded by considering the load factor, improve the power efficiency by turning off the idle nodes and prevent fluctuations around 80% load factor in most case. But these measurements also brings many disadvantages, for example, it only could avoid the overload of a device, but could not guarantee load balancing of the total system; Although turning off idle nodes can save power, restarting these nodes would waste more energy when they are needed again.

Meanwhile, the literatures [15–18] designed some scheduling models for reducing the migration cost of VMs. For example, the literature [15] proposed a migration cost-sensitive load balancing method for the social network multi-agent systems with communities. It is a net profit-based load balancing mechanism, and each load balancing process (i.e., migrating a task from one agent to another one) is associated with a net profit value that depends on the benefit it gains by making a contribution to alleviating the system load unfairness and the cost of migrating the task. [16] leveraged the limited migration costs to save as much energy as possible via dynamic VM consolidation in a heterogeneous cloud datacenter. Considering service delay and migration cost of VMs because of the periodical undergo maintenance processes of the physical nodes, [17] put forward a delay-cost tradeoff VM migration scheme. Apparently, these studies also took into account other goals besides cost when performing VMs migration.

At recent years, as the number of cloud users increase sharply, more problems between users and cloud providers appear. Each client typically has a SLA, which specifies the constraints on QoS that it receives from the system. [19] investigated the SLA-based VM placement and presented an efficient heuristic algorithm based on convex optimization and dynamic programming to minimize the total energy cost of cloud computing system while meeting the specified client-level SLAs in a probabilistic sense, and the monitoring VM is semi-static process in this paper. To guarantee SLAs across a range of QoS requirements, [20] utilized data splitting-based query admission and resource scheduling to propose the admission control and resource scheduling algorithms for Analytics-as-a-Service (AaaS) platforms, which could maximize profits while providing the time-minimized query execution plans to meet user demands and expectations. [21] aimed at supporting research in QoS management by providing a survey of the state of the art of QoS modeling approaches suitable for cloud systems, and developed a co-clustering algorithm to find servers that have similar workload pattern for increasing the running time. Based on software-defined networking, [13] proposed a novel framework to fulfill the QoS requirements of various IoT services and to bal-

ance traffic between IoT servers simultaneously. a QoS-aware VM scheduling method for energy conservation is presented in [22] to meet the QoS requirements.

The above scheduling methods were based on the relations between requests and VMs. That is to say, they realized efficient VM scheduling based on tasks running on them and but do not consider the actual situation of physical nodes, which may make the results not meet the real needs of networks. It is another point for the resource scheduling between VMs and physical nodes. Besides, these studies aimed at either load balancing or migration cost under certain response time or energy efficiency, which can not meet true balance of supply and demand between users and cloud providers.

3 BIG DATA CENTERS MODELING AND PROBLEM STATEMENT

3.1 System Model

This paper focuses on the scheduling between VMs and physical nodes rather than the scheduling between tasks and VMs. For ease of understanding, this paper only chooses a cluster in big data centers as our studied model and gives the target problem statement. The mapping between VMs and physical nodes in the cloud cluster is described in Fig.1.

In the cluster in Fig.1, there are I physical nodes and J VMs. Each physical node can host several VMs. The symbol $Node_i$ means the i -th physical node, and $Node_0, Node_1, Node_2, \dots, Node_{I-1}$ denotes all physical nodes in this cluster. VM_j denotes the j -th VM and the total VMs are $VM_0, VM_1, VM_2, \dots, VM_{J-1}$. $U_{j,i}$ represents the utilization rate of VM_j that is hosted in a corresponding physical node $Node_i$. In Fig. 1, the VM_0, VM_1, VM_6 are allocated in $Node_0$; Both $Node_1$ and $Node_{I-2}$ only contain one VM that are VM_{13} and VM_{J-1} respectively; In $Node_2$ and $Node_{I-1}$, there are merely two VMs, respectively. For physical nodes and VMs, there are many different allocation strategies. Suppose that the set of allocation strategy between VMs and physical nodes is $S = \{s_0, s_1, \dots, s_m\}$, the paper aims at finding the best strategy in S to optimize resource utilization, meet load balancing, minimize migration cost and get the goals of SLA.

3.2 Load of a VM or Physical Node

For a VM or physical node, CPU and memory are two important resources that are often used to measure its load or utilization in the construction of scheduling models, such as the literatures [14, 22, 25–37]. Because

Fig. 1 The example architecture of a cluster in cloud.

of the different design aims of the different scheduling algorithms in these works, in addition of CPU and/or memory, some factors such as network bandwidth, disk, RAM, compute and I/O, sometimes are also considered as one of resource measurement indicators of a VM or physical node. For example, [30] only used the CPU utilization to construct physical machines' average power consumption model for developing energy efficient VM scheduling. [31] leveraged VMs' memory to sense the cache behaviour data of physical machines when designing memory-aware scheduling strategy for cyber-physical systems. [33] adopted CPU and memory to measure the processing capability of VMs or physical nodes. In [22], memory and network bandwidth were used to measure the load of a physical node. [36] leveraged RAM, storage and bandwidth to evaluate the resource utilization of a physical node. CPU, memory and storage are three factors to measure the capacity of each physical server when building a non-preemptive VM scheduling model in [37]. Apparently, when handling the scheduling problem, all shared resources can be seen as the measurement indicators to construct or extend the corresponding model functions according to the actual design purpose of scheduling. That is to say,

they can be also chosen as measurement factors to measure the utilization of a VM or physical node when resources like cache and controllers are also shared. Based on the above analysis and actual needs, this paper uses CPU, network bandwidth and memory to measure the load of a VM or physical node, and define a new metric that is the production of CPU, network bandwidth and memory as the comprehensive load measurement for a virtual server according to the literatures [38] and [39]. The calculation of the load on VM_j is shown in Equation (1).

$$L_{VM_j} = \frac{1}{1 - CPU_{U_{j,i}}} \times \frac{1}{1 - NET_{U_{j,i}}} \times \frac{1}{1 - MEM_{U_{j,i}}} \quad (1)$$

where L_{VM_j} denotes the load of VM_j ; $CPU_{U_{j,i}}$, $NET_{U_{j,i}}$ and $MEM_{U_{j,i}}$ respectively represents the utilization rate of its CPU, network bandwidth and memory. Apparently, the bigger the value of L_{VM_j} is, the higher its load.

In terms of VM, its load information is recorded by the monitor of Eucalyptus and contains CPU, network bandwidth and memory in 60 seconds. Because a physical node can virtualize many VMs, we can simply add

the capacity of the whole VMs hosted in the node. Suppose that $\overline{L_{VM_j}}$ is the average load of VM_j on $Node_i$, and then the overall load of the i -th physical node LN_i is calculated according to Equation (2).

$$LN_i = \sum_{VM_j \in Node_i} \overline{L_{VM_j}} \quad (2)$$

3.3 Load Balancing between Physical Nodes

Known from the above section, the average load of each physical node in 60 seconds can be obtained. If there is I physical nodes in a cluster, then the whole load of the system SUM_{LN} is shown in Equation (3).

$$SUM_{LN} = \sum_{i=0}^I LN_i \quad (3)$$

Without considering other constraints, we can gain the optimal allocation strategy of physical nodes so long as SUM_{LN} reaches the maximum value. Whereas, in fact, when the resource utilization of a physical node is big, the system load may be over, and the performance of cloud services would also reduce. Therefore, how to balance the load with maximum resource utilization is quite necessary. This paper considers using the standard deviation of the load of physical nodes to constrain the obtaining of the maximum load for the system. Assume that the mean load of the i -th physical node is $\overline{LN_i}$, then, the standard deviation of all physical nodes is as follow.

$$\sigma = \sqrt{\frac{1}{I+1} \sum_{i=0}^I (LN_i - \overline{LN_i})^2} \quad (4)$$

where σ denotes the standard deviation of the system load.

Making the full use of the limited physical nodes refers to getting the highest resource utilization while realizing load balancing. That is to say, the smaller σ and the bigger SUM_{LN} can make the system performance be better. To facilitate the measurement of load balancing indicators, this paper adopts TOPSIS [40] to achieve the integration of the above two factors, and it is shown in Equation (5).

$$\varphi = \sqrt{(SUM_{LN} - SUM_{\max})^2 + (\sigma - \sigma_{\min})^2} \quad (5)$$

where $SUM_{\max}$ represents the maximum value of the overall system load, and $\sigma_{\min}$ denotes the minimum value of the standard deviation of all physical nodes. Obviously, when φ gets the minimum value, the load on physical nodes is balanced, and the system performance will be the best.

3.4 Migration Cost of VMs

As everyone knows, migrating VMs will delay the response time of the whole system and reduce the performance of tasks running on them, and namely, there is a price to pay for VM migration. During big data cloud services, the migration ways of VMs usually include P2V(Physical to Virtual), V2V(Virtual to Virtual) and V2P(Virtual to Physical), and this paper uses the V2V method that has offline(static) migration and online(live) migration. The former needs to shut down VMs and copy the system to the goal host, and has an obvious unavailable time, which is not fit for the current big data network environment with great demand change. This paper selects the latter to achieve load balancing and meet the needs of users because it has extremely short unavailable time for migrating VMs. Its migration process accepts the pre-copy way and is shown in Fig.2.

In Fig.2, the two physical nodes have their own existing data, meanwhile, the data of VM_1 on physical node 1 needs to be migrated to physical node 2, which includes three steps. *Step 1*: the migration begins, but the VM_1 continues to run on the physical node 1 and the physical node 2 doesn't start to execute the migration. That is to say, the memory of VM_1 does not copy to the goal node. *Step 2*: VM_1 in physical node 1 sends the whole memory data to the goal VM in physical node 2 by a loop. After copying, the tasks in VM_1 in physical node 1 continue to run and create some new data. *Step 3*: the remaining loops are responsible to send these new data to the goal VM, and the migration finishes.

Even though the downtime of live migration is little, it has a negative influence on the performance of tasks in the VMs. [41] studied the influence and found that the performance reduction while migrating VMs depended on the amount of pages of updating disk during its execution. Therefore, based on [41], this paper estimates the performance reduction as 8% of CPU utilization rate, and then the negative influence during the migration is calculated according to Equation (6).

$$\begin{cases} C_{j,i} = \int_{t_0}^{t_0+t_{j,i}^u} U_{j,i}(t) dt \times 8\% \\ t_{j,i}^u = \frac{M_{j,i}}{B_{j,i}} \end{cases} \quad (6)$$

where $C_{j,i}$ denotes the cost of the whole performance reduction when migrating the j -th VM to $Node_i$; t_0 means the start time of migration and $t_{j,i}^u$ is the migration time of VM_j ; $M_{j,i}$ represents the size of memory of $Node_i$ occupied by VM_j and $B_{j,i}$ means the network bandwidth of $Node_i$ providing to VM_j ; $U_{j,i}(t)$ is the CPU utilization rate of $Node_i$ that executes VM_j . Obviously, the smaller $C_{j,i}$, the better the system performance.

Fig. 2 The steps of pre-copy.

In fact, the whole migration cost should contain the performance reduction of migration process and the benefit after migrating, and the latter mainly focuses on the degree of load balancing of physical nodes. This paper uses the ratio of the standard deviation σ^* after migrating and the standard deviation σ before migrating to evaluate the benefit after migrating. Suppose the average benefit after migrating is BF_i , then,

$$BF_i = 1 - \frac{\sigma^*}{\sigma} \quad (7)$$

When migrating VMs achieves the better load balancing, σ^* is smaller and BF is bigger. Therefore, the whole migration cost MC can be evaluated via using the difference between the performance reduction of

VMs and the migration benefit of the system, namely,

$$MC = \partial \times \sum_{i=0}^I \sum_{j=0}^J C_{j,i} - \sum_{i=0}^I BF_i \quad (8)$$

where ∂ is an adjust parameter and can make the cost lower and the benefit higher when calculating the best migration strategy. Obviously, the better the system performance or the allocation strategy of VMs, the smaller the value of MC .

3.5 QoS Evaluation

Usually, the evaluation indexes of QoS in big data centers contain the downtime and the resource utilization rate of all physical nodes. [22] defined the downtime as the sum of the access time of the log files and switch or migration time. This paper also considers exchanging or transferring data while reading logs. For a VM allocation strategy, suppose that the memory image is transmitted X times; DT represents the downtime, and L^x is the access time of all the remaining log files when the log file transfers at x time. Then,

$$\begin{cases} DT = \sum_{x=1}^X L^x + \sum_{i=0}^I \sum_{j=0}^J t_{j,i}^u \\ L^x = \sum_{i=0}^I \sum_{j=0}^J H_{j,i} \times \frac{D_{VM_j}}{B_{j,i}} \end{cases} \quad (9)$$

where $H_{j,i}$ is the flag to judge whether VM_j is hosted on $Node_i$, if it is, then $H_{j,i} = 1$, otherwise $H_{j,i} = 0$; D_{VM_j} represents the size of dirty pages transferred by VM_j . To keep the consistency of VM memory conditions before and after the migration, the dirty pages produced in the process of memory transmission are sent to the goal physical node during the next transmission. Hence, D_{VM_j} can be set to the size of the mirror memory of VM_j when $x = 0$, $D_{VM_j} = \eta \times L^{x-1}$ when $x \geq 1$, and η is the producing rate of memory dirty page.

The resource utilization rate of all physical nodes is equal to the ratio of the utilization of VMs running on all physical nodes to the number of physical nodes running, namely,

$$RU = \frac{\sum_{i=0}^I \sum_{j=0}^J U_{j,i}}{\sum_{i=0}^I k_i} \quad (10)$$

where RU represents the resource utilization rate of the whole physical system; k_i is the flag to judge whether the i -th physical node is running, and $k_i = 1$ when some tasks are running on VMs hosted on $Node_i$, otherwise $k_i = 0$. $U_{j,i}$ means the utilization rate of $Node_i$ that executes VM_j , and is proportional to CPU, memory and network bandwidth. Therefore, this paper let $U_{j,i}$ be equal to the sum of utilization of the above three

indicators when VM_j is running on $Node_i$, namely $U_{j,i} = CPU_{U_{j,i}} + NET_{U_{j,i}} + MEM_{U_{j,i}}$, otherwise $U_{j,i} = 0$.

In real applications, the minimum downtime and the maximum resource utilization rate are required. That is to say, for an allocation strategy, the shorter the downtime DT and the bigger the resource utilization rate RU , the faster the system processes. Therefore, QoS must meet the following conditions.

$$\min(DT) \ \& \ \max(RU) \quad (11)$$

Obviously, when the downtime or the resource utilization rate is violated, the schedule algorithm should reallocate right VMs.

4 QMOD: QOS-AWARE AND MULTI-OBJECTIVE DYNAMIC VM SCHEDULING

The paper actually focuses on the multi-objective problem, and the usual concerns of the resource scheduling in cloud computing are load balancing, migration cost and QoS. GA is an intelligent optimization algorithm based on natural evolution and selection mechanism to solve multi-objective problem. It has strong global optimization ability and can gain the global optimization solution in the Pareto sense. GA could evolve a population of candidate solutions to a given problem via using operators inspired by natural genetic variation and natural selection. Each solution strategy of the problem is an individual that is also called chromosome, and many individual constitutes a population. In this paper, one allocation strategy of VMs is an individual and a group of allocation strategies consist of a population. That is to say, different strategies (how to allocate VMs to physical nodes) compose a population in GA. The traditional GA consists of encoding, initialization, fitness function, selection, crossover, mutation and re-inserted. The major work in the section is designing encoding method, fitness function, selection, and crossover methods. Because of the special situation in cloud computing, some steps of the traditional GA should be improved to adapt the environment of big data centers in clouds, and the detailed steps are illustrated as follows.

4.1 Encoding Method and Population Initialization

Because the common binary encoding is unnatural and unwieldy for many problems, the paper adopts the array coding method that can fully show the mapping relation between VMs and physical nodes. In the array, the elements' position indicates the label of the VMs

and the elements' value represents the serial number of corresponding physical nodes that host the VMs in the chromosome. As shown in the following example.

$$\begin{bmatrix} 0 & 1 & 2 & 3 & \cdots & J-3 & J-2 & J-1 \\ 2 & 1 & I-1 & 0 & \cdots & 1 & 3 & I-3 \end{bmatrix}$$

In the example chromosome, the amount of VMs and physical nodes is respectively J and I . The first line means the label of the VMs and does not show up in real encoding array. Every VM only runs on a physical node and but a physical node can run several VMs. That is to say, the relation between the physical nodes and VMs is the one-to-many mapping. For example, both VM_1 and VM_{J-3} are allocated to $Node_1$. In fact, each chromosome is one kind of allocation strategy, and a population consists of many chromosomes. For example,

$$\begin{bmatrix} 0 & I-3 & 6 & 2 & \cdots & I-1 & I-4 & 2 \\ 3 & 2 & 6 & 1 & \cdots & j & 2 & 6 \\ 8 & 4 & 9 & 5 & \cdots & 2 & 5 & 8 \\ 2 & 7 & 5 & 2 & \cdots & 6 & 6 & 1 \\ \vdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots \\ I-1 & I-1 & 4 & 7 & \cdots & I-2 & I-3 & 0 \\ 2 & 4 & 7 & 0 & \cdots & 3 & 3 & 2 \\ 5 & 2 & 3 & 7 & \cdots & 5 & 6 & 0 \end{bmatrix}$$

Each row represents one sort of chromosome. A number of chromosomes compose one kind of population. The above example array can be a result of one iteration. To get the globally optimal solution, the original population should be given in advance based on the reported history data of the VMs allocation strategy and the current situation of large changes in VMs load. Obviously, if there is forty chromosomes in a population, then the population can be defined as follows.

$$s_0, s_1, s_2, \cdots, s_{37}, s_{38}, s_{39}$$

Because the final solution must be a globally optimal result, the initialization should be randomly generated to make the solutions evenly distributed to the solution space. That is to say, if there are 40 chromosomes in one population, 8 physical nodes and 14 VMs, the generated matrix in this step should be a matrix with 40 rows and 14 columns. In addition, the process may produce some unavailable results, but these results will be filtered out according to the fitness function.

4.2 Multi-Objective Fitness Function

The reasonable fitness function is an important step to find the optimal allocation strategy. This paper focuses

on the load balancing of physical nodes and the minimum migration cost of VMs when QoS of cloud is guaranteed. First, the precondition of load balancing is to get the highest utilization of physical nodes. Second, the migration of VMs between physical nodes needs to pay cost for the performance reduction, but benefit for the better load balancing among physical nodes. It means that the migration cost of VMs includes the performance reduction and the benefit of the migration. Third, no matter which cloud service, it must reach a certain QoS. Thus, the objective function is to minimize the load balancing of system and the migration cost of VMs under the conditions of minimum the downtime and maximum the resource utilization of physical nodes from the equations (5), (8) and (11).

As there are a large number of VM migrations during task execution, the joint optimized utility function and objective function are necessary for the decision of the final optimized VM scheduling strategies [14]. Multiple Criteria Decision Making (MCDM) and Simple Additive Weighting (SAW) are powerful techniques to achieve the goal of joint optimization of load balancing, migration cost and QoS. However, the measurement units of the above three indicators are different obviously. Therefore, to simplify the calculation of the objective function when the QoS measured by the downtime and the resource utilization rate is met as much as possible, the paper adopts the normalization method to integrate them and express the fitness function F .

$$F = \alpha \times \frac{\varphi - \varphi_{\min}}{\varphi_{\max} - \varphi_{\min}} + \beta \times \frac{MC - MC_{\min}}{MC_{\max} - MC_{\min}} \quad s.t.$$

$$\min(DT) \ \& \ \max(RU) \quad (12)$$

where α and β ($0 \leq \alpha, \beta \leq 1, \alpha + \beta = 1$) are the weight of load balancing and migration cost respectively. $\varphi_{\max}$ and $\varphi_{\min}$ are the maximum and minimum values of the load balancing of physical nodes at a certain time. $MC_{\max}$ and $MC_{\min}$ are the maximum and minimum values of the migration cost of VMs at a certain time. Besides, F is the utility value and is used to exclude chromosomes that do not meet the QoS, so the chromosomes of a population can be sorted according to F to select the good individuals. The smaller the value of F , the better the results.

4.3 Selection of Chromosomes

The purpose of selection process is to screen these good chromosomes from the results processed by the fitness function. For those chromosomes in the population, the fitter chromosomes produce more offspring than the less

Fig. 3 Roulette Wheel Selection.

fit ones. That is to say, the fitter the chromosome, the more times it is likely to be selected to reproduce. The paper adopts the Roulette Wheel Selection method that is conceptually equivalent to giving each individual a slice of a circular roulette wheel equal in area to the individuals fitness, and the roulette wheel can be represented as Fig.3.

In Fig.3, P_{S_i} represents the selection probability of allocation strategy S_i , and every allocation strategy is one chromosome. Apparently, the selection probability is related with the fitness, and the probability of an individual is the individuals' fitness divided by the whole fitness of the population, and then,

$$P_{S_i} = \frac{F_i}{\sum_{i=1}^n F_i} \quad (13)$$

where n is the number of individual chromosomes, namely, the wheel is spun n times. On each spin, the individuals under the wheels marker are selected to be in the pool of parents for the next generation. The cumulative probability CP_{S_i} is the sum of the probability of previous chromosomes and current chromosome, and is expressed as follows,

$$CP_{S_i} = \sum_{q=1}^i P_{S_q} \quad (14)$$

Actually, every time the algorithm randomly generates one number r ($0 \leq r \leq 1$). The i -th chromosome is selected if CP_{S_i} meets the following conditions.

$$CP_{S_{i-1}} \leq r \leq CP_{S_i} \quad (15)$$

Usually, an individual is repeatedly selected as a parent chromosome with probability proportional to its fitness. However, in some cases, some individuals which fitness and selection probability are both low should be chosen rather than directly discarded because their offspring may have high fitness. Thus, those chromosomes with low fitness can be passed on by the Roulette Wheel Selection in this paper.

Fig. 4 An example of the crossover process based on tree model.

4.4 Crossover and Mutation

The crossover operation exchanges subparts of two chromosomes to generate two offspring. The common method of crossover is the single point crossover that is appropriate for binary encoding. If this paper uses the single point crossover, then the exchanged chromosomes are more likely to be unavailable in the actual situation because some VMs may be moved to unavailable physical nodes. Therefore, the paper proposes tree crossover method to solve the above problem. An example of the crossover based on tree model is shown in Fig.4.

In Fig.4, the root of the tree is the cluster controller in big data cloud center, the nodes in the second floor are physical nodes, and the leaf nodes denote VMs. Every chromosome has 3 physical nodes and 5 VMs, and the first chromosome is (0, 0, 1, 1, 2) and the second chromosome is (0, 1, 0, 1, 2). This paper sets the crossover probability P_c as 0.15 and the operation includes four steps. First, the two chromosomes are ready to cross. Second, the VMs that are corresponding to more than two physical nodes need to be picked out. According to the fitness function in the third step, these selected VMs are allocated to the available physical nodes in the fourth step.

While the crossover is the major instrument of variation and innovation in GAs, the mutation prevents the loss of the diversity at a given position. The mutation operation mutates each locus of chromosomes with probability P_m , and places the resulting chromosomes in the new population. In the paper, the mutation probability P_m is 0.01. For the above example, there are 40 chromosomes and every chromosome that is an array has 14 integers, and then the expected number of mutation locus is $40 \times 14 \times 0.01 = 5.6$. Finally, the mutation replaces the current population with the new population.

4.5 QMOD Scheduling Algorithm

After the improved conditions and functions of GA are set, the corresponding algorithm should be given. The paper assumes that the maximum iteration G of the algorithm is 200, which makes many results have been convergent. The key pseudo code of the algorithm is as follows.

Compared with the traditional GA, our improved scheduling algorithm has several advantages. First, QMOD model is a dynamic scheduling scheme based on the real-time situation of VMs and physical nodes. Second, it leverages the tree model to build the relation-

Algorithm 1 QMOD VM Scheduling

Input: VM set $\{VM_j\}$;
Physical Node set $\{Node_i\}$;
Output: Optimal allocation strategy S_p .

```

1   $P_c=0.15, P_m=0.01, G=0, M_{40 \times 14}=\text{null}, p=0$ ;
2   $Pop = \{S_0, S_1, S_2, \dots, S_{37}, S_{38}, S_{39}\}$ ;
3  while  $G \leq 200$  do
4    calculate  $F_p$  for every element in  $Pop$ ;
5     $newPop = \text{null}$ ;
6    while  $p \leq 39$  do
7      select two individuals  $S_m, S_n$  from  $Pop$  according
      to Equations (12)-(15);
8      if (  $\text{random}(0,1) < P_c$  )
9         $S_m$  and  $S_n$  cross based on the steps in Fig.4.
10     endif
11     if (  $\text{random}(0,1) < P_m$  )
12        $S_m$  and  $S_n$  mutate according to  $P_m$ ;
13     endif
14     join  $S_m$  and  $S_n$  into  $newPop$ ;
15   endwhile
16    $Pop = newPop$ ;
17 endwhile
18 return  $S_p$  with the maximum  $F$  meeting QoS condi-
    tions in  $Pop$ ;
```

ships between nodes in cloud clusters when running the crossover operations, which can effectively avoid VMs to be allocated to unavailable physical nodes. Finally, QMOD adopts three indicators to construct the fitness function, which not only guarantee the QoS of cloud services and the load balancing of cloud infrastructures, but also the migration cost of VMs. After executing Algorithm 1, an optimal solution of the VMs allocation can be obtained.

5 PERFORMANCE ANALYSIS AND DISCUSSION

5.1 Simulation Settings and Comparison Algorithms

In this section, we first set up a cloud platform, namely Eucalyptus, and use the Managed (No-VLAN) model. Second, the 4 physical nodes are open with CentOS 8.0, and there are 8 VMs are hosted in them. The cloud platform will report the status of VMs every 60 seconds. Third, we use the datasets with diverse characteristics to validate the efficiency and robustness of the proposed scheme via comparing with Greedy algorithm, Round Robin algorithm and NSGA II having existed in Eucalyptus. The three algorithms based on different approaches provide resource scheduling cases to study and compare the behavior of the proposed scheme. This paper mainly takes the datasets with two kind of initial statuses of the VMs as the experimental data, which are shown in TABLE I and TABLE II, respectively.

According to the data in TABLE I and TABLE II, the initial allocation strategies of VMs can be obtained,

Table 1 Initial Status 1 of Virtual Machine.

Virtual Machine	Utilization		
	CPU	Net Bandwidth	Memory
VM_0	46%	30%	26%
VM_1	42%	23%	33%
VM_2	60%	57%	25%
VM_3	74%	28%	47%
VM_4	41%	37%	34%
VM_5	59%	45%	56%
VM_6	70%	75%	60%
VM_7	52%	58%	73%

Table 2 Initial Status 2 of Virtual Machine.

Virtual Machine	Utilization		
	CPU	Net Bandwidth	Memory
VM_0	48%	53%	64%
VM_1	34%	20%	39%
VM_2	42%	38%	56%
VM_3	52%	47%	59%
VM_4	43%	32%	28%
VM_5	23%	20%	24%
VM_6	24%	26%	34%
VM_7	32%	45%	53%

for example, (2, 0, 1, 4, 2, 3, 4, 1) and (1, 4, 2, 3, 3, 0, 2, 0) are two initial chromosomes for TABLE I. In each of the next 60 seconds, the change of load in VMs and physical nodes can be read from the system records. In addition, the paper evaluates the performance of our scheduling scheme in terms of the utilization of physical nodes, the standard deviation of load in physical nodes, the performance reduction of load and the convergence of solution.

5.2 Utilization of Physical Nodes

The load of physical nodes is the sum of load of VMs hosted in them. In the real big data centers, the number of physical nodes is far less than that of cloud services. Therefore, to better make use of the limited physical resource, the utilization of physical nodes needs to be maximized. That is to say, the higher the utilization of physical nodes is, the better the performance of cloud services is. We compare our proposed scheme with the other three algorithms in term of the utilization of physical nodes, and the comparison results under different VMs status are shown in Fig.5.

Seen from the difference of the four algorithms in Fig.5(a) and Fig.5(b), regardless of the Runtime or the initial status of VMs, the utilization of physical nodes is high and stable all the time via using our proposed scheduling scheme, although the performance of NSGA II is sometimes better than our model. Besides, Greedy algorithm has a high utilization rate only at the beginning, but with the passage of time, its resource utilization gradually decreases. Meanwhile, Round Robin

Fig. 5 Comparison of all four algorithms in terms of the utilization of physical nodes based on initial status in (a) TABLE I and (b) TABLE II, respectively.

algorithm always has a low resource utilization and has no large fluctuation all the time. In short, based on the values and fluctuation of utilization of physical nodes, our proposed scheduling scheme can apparently make full use of the limited physical resource, which means that the higher and more stable cloud services can be provided.

5.3 Load Balancing between Physical Nodes

Besides the high utilization of physical nodes, the load balancing among them is also very important for cloud services. This paper designs the proposed VM scheduling scheme via leveraging a multi-objective fitness function to achieve both the high utilization and load balancing of physical nodes, and the standard deviation is used to evaluate the degree of balancing. The comparison results of all four modes are described in Fig.6 in terms of the load balancing performance of physical nodes changing with the runtime.

Fig. 6 Comparison of the standard deviation of resource utilization for the four models based on initial status in (a) TABLE I and (b) TABLE II, respectively.

In Fig.6, no matter which status, our proposed scheduling scheme always gains the low standard deviation of load. Nevertheless, the standard deviation results of load with NSGA II are crossed with the results of our proposed model and Round Robin algorithm, and the overall performance is between them. The standard deviation of load with Greedy algorithm is high and becoming bigger with time goes on, and the standard deviation of Round Robin has little fluctuation and is not high. The main cause is, based on the average load of physical nodes, the difference between the average load and every load for a physical node can be obtained in our proposed scheduling scheme, which means that our scheme reduces the number of idle VMs, wastes less resources, and achieves high and stable resource utilization. Greedy algorithm used the next node to complete the tasks until the previous node is exhausted, which means that the difference between physical nodes is large and the situation is becoming worse. Round Robin evenly allocated VMs to physical nodes in one loop, so the load balancing of physical nodes with Round

Robin is relatively small. Meanwhile, because NSGA II can obtain relatively excellent individuals by using elite strategy, the load of the system is more stable than the latter two. As the above-mentioned results, our proposed scheme can better achieve load balancing than the other three.

5.4 Evaluation of Migration Cost

In the paper, the migration cost is evaluated by the difference between the performance reduction and the benefit of migration. The performance reduction during the process of migration can be reported by monitoring and the benefit of every migration can be calculated with the change of load balancing. According to the reported data about the performance of load, the mentioned three schemes have their own performance reduction with different total workloads. The corresponding comparison results are shown in Fig.7 for the migration cost of the three schemes.

Fig. 7 Comparison of the performance reduction of load based on initial status in (a) TABLE I and (b) TABLE II, respectively.

Known from Fig.7, regardless of the load situation of physical nodes, our proposed scheduling scheme is still the one with the least decline in terms of the performance reduction of system than the other three. It is obvious that the difference between NSGA II and our scheme is the least for the evaluation of performance reduction, and especially the difference does not exist nearly with the continue adding of the system loads and even emerges the reverse. Meanwhile, Round Robin still performs better than Greedy algorithm, too. The main cause is, Greedy only focused on the availability of physical nodes during the migration, but not considered the migration cost. However, Round Robin could select the proper physical nodes based on VMs evenly distributed in physical nodes. To put into a nutshell, our proposed scheduling scheme has relatively little migration cost.

5.5 Convergence of Solution

The convergence is an important index to judge whether an algorithm is reasonable and accurate. Usually, the mean of a population can be clearly represented as the change of the simulation results. When the mean does not fluctuate, we can say that the algorithm converges and the optimal solution is obtained. That is to say, the convergence is leveraged to get the optimal solution and trace the status of results, and the simulation results can reflect the convergence of the algorithm. In this paper, the iteration of our scheduling algorithm is set to 200, and its convergence is shown in Fig.8.

Fig. 8 Convergence of our proposed scheduling scheme.

In Fig.8, when the iteration reaches about 80, the simulation results of our proposed scheme have no large fluctuation, or the mean of population tends toward stability. That is to say, our scheduling algorithm can be convergent near 80 iterations. Now, the maximum value of the convergence is 6.473. Apparently, the convergence result proves the feasibility of our proposed scheduling scheme again.

6 OPEN ISSUES AND FUTURE WORK

Our proposed scheme is committed to pay little migration cost as much as possible to keep system load balancing under meeting certain QoS when scheduling VMs dynamically between limited physical nodes in a heterogeneous cloud cluster. Compared to the other models, all numerical results and performance figures have proved the accuracy and effectiveness of our scheme. However, it also has some shortcomings. For example, the fitness of chromosomes is poor and requires a large probability of crossover and mutation in the early evolution, and the chromosome already has a good structure and requires a small probability of crossover and mutation in the later evolution, but this paper does not adopt the variable probability of crossover and mutation. Meanwhile, our scheme does not consider using some measures similar to the elite strategy in NSGA II to accelerate the execution speed of the algorithm, resulting in the disadvantage of being easy to fall into slower search speed in the later stage. Besides, although the proposed scheme based on improved GA has strong global optimization ability, this paper does not verify the optimality of the results in the Pareto sense. In fact, solving these problems can greatly improve the performance of VM scheduling model, and we suggest reflecting them in future study and experimental work.

7 CONCLUDE

To balance the load of big datacenters and improve the QoS of cloud services, this paper proposes a dynamic scheduling scheme between VMs and physical nodes considering load balancing, migration cost and QoS based on the improved genetic algorithm. The array encoding is first proposed to fit for the mapping between physical nodes and VMs. Then, the three different fitness functions are used to select good individual chromosomes. Third, TOPSIS, SAW and the normalization method are adopted to solve the unit difference between the fitness functions. Finally, in the process of crossover, a tree model is proposed to make the crossover more reasonable. The simulation results reveal the good performance of our proposed scheduling scheme.

Acknowledgment

Funding: the National Natural Science Foundation of China (grant number 61907011) and the Key Scientific Research Projects in Colleges and Universities in Henan (grant number 21A520025).

Conflict of Interest: The authors declare that they have no conflict of interest.

Data Availability Statement: The raw/processed data required to reproduce these findings cannot be shared at this time as the data also forms part of an ongoing study.

References

1. Harb H and Makhoul A, "Energy-efficient scheduling strategies for minimizing big data collection in cluster-based sensor networks," *Peer-to-Peer Networking and Applications*, vol.12, no.3, pp.620-634, May 2019.
2. Puthal D, Obaidat M S, Nanda P, Prasad M, Mohanty S P, and Zomaya A Y, "Secure and Sustainable Load Balancing of Edge Data Centers in Fog Computing," *IEEE Communications Magazine*, vol.56, no.5, pp.60-65, May 2018.
3. Srichandan S, Kumar T A, and Bibhudatta S, "Task scheduling for cloud computing using multi-objective hybrid bacteria foraging algorithm," *Future Computing & Informatics Journal*, vol.3, no.2, pp.210-230, Dec. 2018.
4. Li J and Han Y, "A hybrid multi-objective artificial bee colony algorithm for flexible task scheduling problems in cloud computing system," *Cluster Computing*, vol.1, pp.1-17, Dec. 2019.
5. Zhang Y, Deng L, Chen M, and Wang P, "Joint bidding and geographical load balancing for datacenters: Is uncertainty a blessing or a curse?" *IEEE/ACM Trans. Networking*, vol.26, no.3, pp.1049-1062, Apr. 2018.
6. Domanal S G, Guddeti R M R, and Buyya R, "A Hybrid Bio-Inspired Algorithm for Scheduling and Resource Management in Cloud Environment," *IEEE Trans. Services Computing*, vol.13, no.1, pp.3-15, Feb. 2020.
7. Jia M, Liang W, Xu Z, and Huang M, "Cloudlet load balancing in wireless metropolitan area networks," *In Proc. The 35th Annual IEEE International Conference on Computer Communications (INFOCOM 2016)*, Apr. 2016, pp.1-9.
8. Supreeth S and Biradar S, "Scheduling virtual machines for load balancing in cloud computing platform," *International Journal of Science and Research*, vol.2, no.6, pp.437-441, Jun. 2013.
9. Nadeem H A, Elazhary H, Mai A, "Priority-Aware Virtual Machine Selection Algorithm in Dynamic Consolidation," *International Journal of Advanced Computer Science & Applications*, vol.9, no.11, pp.416-420, Jan. 2018.
10. Raj G and Setia S, "Effective Cost Mechanism for Cloudlet Retransmission and Prioritized VM Scheduling Mechanism over Broker Virtual Machine Communication Framework," *International Journal on Cloud Computing: Services and Architecture (IJCCSA)*, Vol.2, No.3, Jun. 2012.
11. Tang F, Yang L T, Tang C, Li J, and Guo M, "A dynamical and load-balanced flow scheduling approach for big data centers in clouds," *IEEE Trans. Cloud Computing*, vol.6, no.4, pp.915-928, Dec. 2018.
12. Yang C T, Chen S T, Liu J C, Su Y W, Puthal D, and Ranjan R, "A predictive load balancing technique for software defined networked cloud services," *Computing*, vol.101, no.3, pp.211-235, Mar. 2019.
13. Montazerolghaem A and Yaghmaee M H, "Load-balanced and QoS-aware Software-defined Internet of Things," *IEEE Internet of Things Journal (Early Access)*, pp.1-1, Jan. 2020.

14. Xu X, Zhang X, Khan M, Dou W, Xue S, and Yu S, "A balanced virtual machine scheduling method for energy-performance trade-offs in cyber-physical cloud systems," *Future Generation Computer Systems*, vol.105, pp.789-799, Apr. 2020.
15. Wang W and Jiang Y, "Migration cost-sensitive load balancing for social networked multiagent systems with communities," *In Proc. IEEE 25th International Conference on Tools with Artificial Intelligence*, Nov. 2013, pp. 127-134.
16. Wu Q, Ishikawa F, Zhu Q, and Xia Y, "Energy and Migration Cost-Aware Dynamic Virtual Machine Consolidation in Heterogeneous Cloud Datacenters," *IEEE Trans. Services Computing*, vol.12, no.4, pp.550-563, Aug. 2019.
17. Wang X, Chen X, Yuen C, et al, "Delay-cost tradeoff for virtual machine migration in cloud data centers," *Journal of Network and Computer Applications*, vol.78, pp.62-72, Jan. 2017.
18. Li J, Li X and Zhang R, "Energy-and-Time-Saving Task Scheduling Based on Improved Genetic Algorithm in Mobile Cloud Computing," *In Proc. International Conference on Collaborative Computing: Networking, Applications and Worksharing*, Nov. 2016, pp.418-428.
19. Goudarzi H, Ghasemazar M, and Pedram M, "SLA-based optimization of power and migration cost in cloud computing," *In Proc. 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, May 2012, pp.172-179.
20. Zhao Y, Calheiros R N, Vasilakos A V, Bailey J, and Sinnott R O, "SLA-Aware and Deadline Constrained Profit Optimization for Cloud Resource Management in Big Data Analytics-as-a-Service Platforms," *In Proc. 2019 IEEE 12th International Conference on Cloud Computing (CLOUD)*, Jul. 2019, pp. 146-155.
21. Ardagna D, Casale G, Ciavotta M, Perez JF, and Wang W, "Quality of service in cloud computing: modeling," *Journal of Internet Services and Applications*, vol.5, no.1, pp.1-17, Sept. 2014.
22. Qi L, Chen Y, Yuan Y, Fu S, Zhang X, and Xu X, "A QoS-aware virtual machine scheduling method for energy conservation in cloud-based cyber-physical systems," *World Wide Web*, vol.23, pp.1275-1297, May 2019.
23. Arroyo J E C and Leung J Y T, "An effective iterated greedy algorithm for scheduling unrelated parallel batch machines with non-identical capacities and unequal ready times," *Computers & Industrial Engineering*, vol.105, pp.84-100, Mar. 2017.
24. Farooq M U, Shakoar A, and Siddique A B, "An Efficient dynamic round robin algorithm for CPU scheduling," *In Proc. IEEE International Conference on Communication, Computing and Digital Systems (C-CODE)*, Mar. 2017, pp.244-248.
25. Yuan M, Li Y, Zhang L, and Pei F, "Research on intelligent workshop resource scheduling method based on improved NSGA-II algorithm," *Robotics and Computer-Integrated Manufacturing*, vol.71, pp.102141, Oct. 2021.
26. Katsalis K, Papaioannou T G, Nikaen N, and Tassioulas L, "SLA-driven VM scheduling in mobile edge computing," *In Proc. IEEE 9th International Conference on Cloud Computing (CLOUD)*, Jun. 2016, pp.750-757.
27. Raghavendra S N, Jogendra K M, and Smitha C C, "A Secured and Effective Load Monitoring and Scheduling Migration VM in Cloud Computing," *IOP Conference Series: Materials Science and Engineering*, Dec. 2020, Vol.981, No.2, pp.022069.
28. Ahmad W, Alam B, Ahuja S, and Malik S, "A dynamic VM provisioning and de-provisioning based cost-efficient deadline-aware scheduling algorithm for Big Data workflow applications in a cloud environment," *Cluster Computing*, vol.24, no.1, pp.249-278, Mar. 2021.
29. Sonkar S K and Kharat M U, "A review on resource allocation and VM scheduling techniques and a model for efficient resource management in cloud computing environment," *In Proc. International Conference on ICT in Business Industry & Government (ICTBIG)*, Nov. 2016, pp.1-7.
30. Wang B, Liu F, and Lin W, "Energy-efficient VM scheduling based on deep reinforcement learning," *Future Generation Computer Systems*, vol.125, pp.616-628, Jul. 2021.
31. Haoxiang W and Smys S, "Secure and optimized cloud-based cyber-physical systems with memory-aware scheduling scheme," *Journal of trends in Computer Science and Smart technology (TCSST)*, vol.2, no.3, pp.141-147, Jul. 2020.
32. Alboaneen D, Tianfield H, Zhang Y, and Pranggono B, "A metaheuristic method for joint task scheduling and virtual machine placement in cloud data centers," *Future Generation Computer Systems*, vol.115, pp.201-212, Feb. 2021.
33. Liang H, Du Y, Gao E, and Sun J, "Cost-driven scheduling of service processes in hybrid cloud with VM deployment and interval-based charging," *Future Generation Computer Systems*, vol.107, pp.351-367, Jun. 2020.
34. Han S W, Min S D, and Lee H M, "Energy efficient VM scheduling for big data processing in cloud computing environments," *Journal of Ambient Intelligence and Humanized Computing*, pp.1-10, Jun. 2019.
35. Cho K M, Tsai P W, Tsai C W, and Yang C S, "A hybrid meta-heuristic algorithm for VM scheduling with load balancing in cloud computing," *Neural Computing and Applications*, vol.26, no.6, pp.1297-1309, Aug. 2015.
36. Ramamoorthy S, Ravikumar G, Saravana Balaji B, Balakrishnan S, and Venkatachalam K, "MCAMO: multi constraint aware multi-objective resource scheduling optimization technique for cloud infrastructure services," *Journal of Ambient Intelligence and Humanized Computing*, vol.12, no.6, pp.5909-5916, Jun. 2021.
37. Psychas K and Ghaderi J, "On non-preemptive VM scheduling in the cloud," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol.1, no.2, pp.1-29, Dec. 2017.
38. Xu H, Liu Y, Wei W, and Xue Y, "Migration Cost and Energy-Aware Virtual Machine Consolidation Under Cloud Environments Considering Remaining Runtime," *International Journal of Parallel Programming*, vol.47, pp.481501, Jun. 2019.
39. Shang Z, Chen W, Ma Q, and Wu B, "Design and implementation of server cluster dynamic load balancing based on OpenFlow," *In Proc. International Joint Conference on Awareness Science and Technology & Ubi-Media Computing*, Nov. 2013, pp.691-697.
40. Abdel-Basset M, Saleh M, Gamal A, and Smarandache F, "An approach of TOPSIS technique for developing supplier selection with group decision making under type-2 neutrosophic number," *Applied Soft Computing*, vol.77, pp.438-452, Apr. 2019.
41. Voorsluys W, Broberg J, Venugopal S, and Buyya R, "Cost of virtual machine live migration in clouds: a performance evaluation," *In Proc. IEEE International Conference on Cloud Computing*, Dec. 2009, pp.254-265.