QoS-aware and multi-objective virtual machine dynamic scheduling for big data centers in clouds

Jirui Li · Rui Zhang · Yafeng Zheng

Accepted: 14 June 2022 / Published online: 25 July 2022 © The Author(s) 2022

Abstract

Efficient resource scheduling is one of the most critical issues for big data centers 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

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 (Harb and Makhoul 2019). Cloud computing is popular for scalable computation and processing of big data to overcome the limitations of individual devices (Puthal et al. 2018). 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, which 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 Puthal et al. (2018), Srichandan et al. (2018), Li and Han (2019), Zhang et al. (2018), Domanal et al. (2020), Jia et al. (2016), Supreeth and Biradar (2013), Nadeem et al. (2018), Raj and Setia (2012), Tang et al. (2018), Yang et al. (2019), Montazerolghaem and Yaghmaee (2020), Xu et al. (2020), Wang and Jiang (2013), Wu et al. (2019), Wang et al. (2017), Li et al. (2016), Goudarzi et al. (2012), Zhao et al. (2019) and Ardagna et al. (2014), 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, Srichandan et al. (2018) 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 Li and Han (2019) 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. Zhang et al. (2018) achieved geographical load balancing via routing workloads dynamically to reduce overall energy consumption. Domal et al. (2020) 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. Jia et al. (2016) 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 studies 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. 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 (Arroyo and Leung 2017), round robin (Farooq et al. 2017) and the non-dominated sorting genetic algorithm (NSGA II) Yuan et al. (2021) 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 Sect. 2, we introduce the previous related work. Section 3 provides the framework of our proposed model and the corresponding problem statement. In Sect. 4, we described the detailed calculations of the proposed algorithm. Section 5 builds the experimental settings and analyzes the performance of our model. Section 6 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 Supreeth and Biradar (2013), Nadeem et al. (2018), Raj and Setia (2012), Tang et al. (2018), Yang et al. (2019), Montazerolghaem and Yaghmaee (2020), Xu et al. (2020) have been proposed. Supreeth and Biradar (2013) 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, Nadeem et al. (2018) and Raj and Setia (2012) 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 bring 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 studies Wang and Jiang (2013), Wu et al. (2019), Wang et al. (2017) and Li et al. (2016) designed some scheduling models for reducing the migration cost of VMs. For example, the literature Wang and Jiang (2013) 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. Wu et al. (2019) leveraged the limited migration costs to save as much energy as possible via dynamic VM consolidation in a heterogeneous cloud data center. Considering service delay and migration cost of VMs because of the periodical undergo maintenance processes of the physical nodes, Wang et al. (2017) put forward a delay–cost trade-off 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. Goudarzi et al. (2012) 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 system. To guarantee SLAs across a range of QoS requirements, Zhao et al. (2019) 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. Ardagna et al. (2014) 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, Montazerolghaem and Yaghmaee (2020) proposed a novel framework to fulfill the QoS requirements of various IoT services and to balance traffic between IoT servers simultaneously. A QoS-aware VM scheduling method for energy conservation is presented in Qi et al. (2019) 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 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 cannot 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 ith physical node, and Node

0

, Node

1

, Node

2

, . . . , Node

I−1

denote all physical nodes in this cluster. VM

j

represents the utilization rate of VM

j

that is hosted in a corresponding physical node Node

j

. In Fig. 1, the VM0, VM1, VM6 are allocated in Node0; both Node1 and Node2 only contain one VM that are VM13 and VM14, respectively; in Node2 and Node4, 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 \( S = \{s_0, s_1, . . . , s_m\} \), 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 studies Xu et al. (2020), Qi et al. (2019), Yuan et al. (2021), Katsalis et al. (2016), Raghavendra et al. (2020), Ahmad et al. (2021), Sonkar and Kharat (2016), Wang et al. (2021), Haoxiang and Smys (2020), Alboaneen et al. (2021), Liang et al. (2020), Han et al. (2019), Cho et al. (2015), Ramamoorthy et al. (2021) and Psychas and Ghaderi (2017). Because 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 the resource measurement indicators of a VM or physical node. For example, Wang et al. (2021) only used the CPU utilization to construct physical machines’ average power consumption model for developing energy efficient VM scheduling. Haoxiang and Smys (2020) leveraged VMs’ memory to sense the cache behavior data of physical machines when designing memory-aware scheduling strategy for cyber-physical systems. Liang et al. (2020) adopted CPU and memory to measure the processing capability of VMs or physical nodes. In Qi et al. (2019), memory and network bandwidth were used to measure the load of a physical node. Ramamoorthy et al. (2021) 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 Psychas and Ghaderi (2017). 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 studies Xu et al. (2019) and Shang et al. (2013).
The calculation of the load on VM $j$ is shown in Eq. (1).

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

where $L_{VM_j}$ denotes the load of VM $j$; and $CPU_{U_{j,i}}$, $NET_{U_{j,i}}$, and $MEM_{U_{j,i}}$, respectively, represent 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 s. Because a physical node can virtualize many VMs, we can simply add the capacity of the whole VMs hosted in the node. Suppose that $LN_{i}$ is the average load of VM $j$ on Node $i$, and then the overall load of the $ith$ physical node $LN_i$ is calculated according to Eq. (2).

$$LN_i = \sum_{VM_j \in Node_i} L_{VM_j}$$

### 3.3 Load balancing between physical nodes

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

$$SUM_{LN} = \sum_{i=0}^{I} LN_i$$

Without considering other constraints, we can gain the optimal allocation strategy of physical nodes so long as $SUM_{LN}$ reaches the maximum value. 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 $ith$ physical node is $LN_i$, then, the standard deviation of all physical nodes is as follows.

$$\sigma = \sqrt{\frac{1}{I+1} \sum_{j=0}^{I} (LN_i - LN_j)^2}$$

where $\sigma$ 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 $\sigma$ and the bigger $SUM_{LN}$ can make the system performance be better. To facilitate the measurement of load balancing indicators, this paper adopts TOPSIS (Abdel-Basset et al. 2019) to achieve the integration of the above two factors, and it is shown in Eq. (5).

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

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 $\varphi$ 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 physical to virtual (P2V), virtual to virtual (2V2) and virtual to physical (2VP), and this paper uses the 2V2 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 does not 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. 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. Voorsluys et al. (2009) 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 Voorsluys et al. (2009), this paper estimates the performance reduction as 8% of CPU utilization rate, and then, the negative influence during the migration is
existing data

\( B_j \) and \( i \) represents the size of memory of Node \( j \) calculated according to Eq. (6).

**Fig. 2** Steps of pre-copy

Calculated according to Eq. (6).

\[
\begin{align*}
C_{j,i} &= \int_{t_0}^{t_{0}+t_{j,i}^u} U_{j,i}(t) \, dt \times 8\
\lambda_{j,i}^u &= \frac{M_{j,i}}{B_{j,i}}
\end{align*}
\]

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 \( j \); and \( U_{j,i}(t) \) is the CPU utilization rate of Node \( i \) that executes VM \( j \). Obviously, the smaller the \( C_{j,i} \), the better the system performance.

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 \( \sigma^* \) after migrating and the standard deviation \( \sigma \) before migrating to evaluate the benefit after migrating. Suppose the average benefit after migrating is \( BF_i \), then,

\[
BF_i = 1 - \frac{\sigma^*}{\sigma}
\]

When migrating VMs achieves the better load balancing, \( \sigma^* \) 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

\[
MC = \delta \times \sum_{i=0}^{J} \sum_{j=0}^{I} C_{j,i} - \sum_{i=0}^{J} BF_i
\]

where \( \delta \) 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. Qi et al. (2019) 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,

\[
\begin{align*}
DT &= \sum_{i=1}^{X} L^x + \sum_{i=0}^{J} \sum_{j=0}^{I} t_{j,i}^u \\
L^x &= \sum_{i=0}^{J} \sum_{j=0}^{I} H_{j,i} \times \frac{D_{VM_j}^d}{B_{j,i}}
\end{align*}
\]

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, \( H_{j,i} = 0 \); \( D_{VM_j}^d \) 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, \( D_{VM_j}^d \) can be set to the size of the mirror memory of VM \( j \) when \( x = 0 \), \( D_{VM_j}^d = \eta \times L^{x-1} \) when \( x \geq 1 \), and \( \eta \) 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,

$$RU = \sum_{i=0}^{J} \sum_{j=0}^{I} U_{j,i} / \sum_{i=0}^{J} k_i$$  \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$_j$, 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.

$$\text{min(DT)} \quad \text{and} \quad \text{max(RU}) \quad \quad (11)$$

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. Genetic algorithm (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.

$$[0 \ 1 \ 2 \ 3 \ ... \ J - 3 \ J - 2 \ 1]$$

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, 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,

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

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.

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

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 Eqs. (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 (Xu et al. 2020). 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 \( F \).

\[
F = \alpha \times \frac{\varphi - \varphi_{\text{min}}}{\varphi_{\text{max}} - \varphi_{\text{min}}} + \beta \times \frac{MC - MC_{\text{min}}}{MC_{\text{max}} - MC_{\text{min}}}
\]

s.t. \text{min}(DT) and \text{max}(RU) \tag{12}

where \( \alpha \) and \( \beta \) (0 ≤ \( \alpha, \beta \) ≤ 1, \( \alpha + \beta = 1 \)) are the weight of load balancing and migration cost, respectively. \( \varphi_{\text{max}} \) and \( \varphi_{\text{min}} \) are the maximum and minimum values of the load balancing of physical nodes at a certain time, \( MC_{\text{max}} \) and \( MC_{\text{min}} \) 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 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 is shown in 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 to the fitness, and the probability of an individual is the individuals’ fitness divided by the whole fitness of the population, and then,

\[
P_{S_i} = \frac{F_i}{\sum_{i=1}^{n} F_i} \tag{13}
\]

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,

\[
CP_{S_i} = \sum_{q=1}^{i} P_{S_q} \tag{14}
\]

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.

\[
CP_{S_{i-1}} \leq r \leq CP_{S_i} \tag{15}
\]

Fig. 3  Roulette wheel selection
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.

### 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 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 the exchanged chromosomes are more likely to 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.

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 pseudocode 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 relationships 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 s. 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 Tables 1 and 2, respectively.

According to the data in Tables 1 and 2, the initial allocation strategies of VMs can be obtained; for example, (2, 0, 1, 4, 2, 3, 4, 1) and (1, 4, 2, 3, 3, 0, 2, 0) are two initial chro-
mosomes for Table 1. In each of the next 60 s, 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. 5a, 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 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.

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 that 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.

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 in particular, 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 that 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.

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 data centers 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.

Funding Funding for this study was received from the National Natural Science Foundation of China (Grant No. 61907011) and the Key Scientific Research Projects in Colleges and Universities in Henan (Grant No. 21A520025).

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

Declarations

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

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

Abdel-Basset M, Saleh M, Gamal A, Smarandache F (2019) An approach of TOPSIS technique for developing supplier selection with group decision making under type-2 neurostrophic number. Appl Soft Comput 77:438–452

Ahmad W, Alam B, Ahuja S, Malik S (2021) A dynamic VM provisioning and de-provisioning based cost-efficient deadline-aware scheduling algorithm for Big Data workflow applications in a cloud environment. Clust Comput 24(1):249–278

Albouneen D, Tianfeld H, Zhang Y, Pranggono B (2021) A metaheuristic method for joint task scheduling and virtual machine placement in cloud data centers. Futur Gener Comput Syst 115:201–212

Ardagna D, Casale G, Ciavotta M, Perez JF, Wang W (2014) Quality of service in cloud computing: modeling. J Internet Serv Appl 5(1):1–17

Arroyo JEC, Leung JYT (2017) An effective iterated greedy algorithm for scheduling unrelated parallel batch machines with non-identical capacities and unequal ready times. Comput Ind Eng 105:84–100

Cho KM, Tsai PW, Tsai CW, Yang CS (2015) A hybrid meta-heuristic algorithm for VM scheduling with load balancing in cloud computing. Neural Comput Appl 26(6):1297–1309

Domanal SG, Guddeti RMR, Buyya R (2020) A hybrid bio-inspired algorithm for scheduling and resource management in cloud environment. IEEE Trans Serv Comput 13(1):3–15

Farooq MU, Shakoor A, Siddique AB (2017) An efficient dynamic round robin algorithm for CPU scheduling. In: Proceedings of IEEE international conference on communication, computing and digital systems (C-CODE), pp 244–248

Goudarzi H, Ghasemazar M, Pedram M (2012) SLA-based optimization of power and migration cost in cloud computing. In: Proceedings of 12th IEEE/ACM international symposium on cluster, cloud and grid computing, pp 172–179

Han SW, Min SD, Lee HM (2019) Energy efficient VM scheduling for big data processing in cloud computing environments. J Ambient Intell Human Comput. https://doi.org/10.1007/s12652-019-01361-8

Haoxiang W, Smys S (2020) Secure and optimized cloud-based cyber-physical systems with memory-aware scheduling scheme. J Trends Comput Sci Smart Technol (TCSST) 2(3):141–147

Harb H, Makhoul A (2019) Energy-efficient scheduling strategies for minimizing big data collection in cluster-based sensor networks. Peer-to-Peer Netw Appl 12(3):620–634
