Hindawi
Mathematical Problems in Engineering
Volume 2021, Article ID 5575129, 15 pages
https://doi.org/10.1155/2021/5575129

Research Article

LBPSGORA: Create Load Balancing with Particle Swarm Genetic Optimization Algorithm to Improve Resource Allocation and Energy Consumption in Clouds Networks

Seyedeh Maedeh Mirmohseni, Amir Javadpour, and Chunming Tang

School of Mathematics and Computer Science, Guangzhou University, Guangzhou 510006, China

Correspondence should be addressed to Amir Javadpour; a_javadpour@e.gzhu.edu.cn and Chunming Tang; ctang@gzhu.edu.cn

Received 21 January 2021; Revised 17 March 2021; Accepted 20 May 2021; Published 7 June 2021

Academic Editor: Nazrul Islam

Due to the purpose of this study that reducing power consumption in the cloud network is based on load balancing, the fitness function measures the load balance between cloud network and servers (the hosts). This technique is appropriate for handling the resource optimization challenges, due to the ability to convert the load balancing problem into an optimization problem (reducing imbalance cost). In this research, combining the results of the particle swarm genetic optimization (PSGO) algorithm and using a combination of advantages of these two algorithms lead to the improvement of the results and introducing a suitable solution for load balancing operation, because in the proposed approach (LBPSGORA), instead of randomly assigning the initial population in the genetic algorithm, the best result is procured by putting the initial population. The LBPSGORA method is compared with PSO, GA, and hybrid GA-PSO. The execution cost, load balancing, and makespan have been evaluated and our method has performed better than similar methods.

1. Introduction

Recent advances in computer networks, devices, Internet speeds, and so forth have led many companies and users to use cloud services. Resource sharing, virtualization of various services, and scheduling techniques have increased the efficiency of data centers. All of this has led users to use cloud services to meet their needs. Users need strong processors and fast networks to set up cloud data centers in different parts of the world. One of the most important concerns in this area is related to energy [1, 2]. For this purpose, in recent years, attention has been paid to enhanced techniques for providing power management for cloud servers. Therefore, many researchers have worked on different algorithms to reduce costs by reducing energy consumption and also reduce greenhouse gas emissions. To improve the performance of several algorithms to affect the hardware or the software has been tested. With the help of virtualization techniques, several virtual machines can be created on a physical machine, so we need less hardware and can increase performance. Cloud computing uses virtualization technology based on user's needs to provide resources, which reduces energy consumption. One of the most popular ways to reduce energy consumption is to use metaheuristic techniques [3, 4]. This technique could attract attention due to its simplicity of implementation. The cloud services providers, because of the benefits they gain from serving customers as well as cloud consumers, such as organizations and economic organizations, tend to cloud computing because of the benefits they receive from reducing or eliminating infrastructure maintenance costs for these services [1]. However, since applications for business operations of users running in the cloud may be highly sensitive, consumers need to have a guarantee by the cloud providers when the service is delivering. Usually, such a guarantee is supplied between the consumer and the provider through a service level agreement. Recently, the use of virtualization to improve the performance of cloud networks has flourished [2, 5]. So virtual machines can be moved from one host to another to enhance the efficiency of users' applications without interrupting service. Also, it is possible that the

number of resources allocated to a customer increases or decreases dynamically [6–9].

In cloud computing, the users pay the cost for the amount of use and services they receive, and the workspace is transmitted to the cloud (the Internet) from data and information on the personal computer [10] so users can access their data and evidence anytime, anywhere in the world. Users also do not need hardware systems with high processing capacity and storage capacity, as all cloud computing operations and storage are handled in the cloud service providers section by equipped and advanced servers [11]. Currently, cloud computing services are offered in three different forms: SaaS_Software as a Service, PaaS_-Platform as a Service, and IaaS_Infrastructure as a Service [1]. Organizations and companies can select and use one or more service strategies depending on their needs. Cloud accounting is one of the most famous words in today's economic organization. An intrinsic cloud feature that distinguishes it from traditional and old hosting services. The amount of unlimited resource capacity (such as CPUs, memory, input-output network, and disk) is provided at a competitive rate [12]. This subject provides opportunities for startup companies to host their applications in the cloud, so it eliminates the overflow of prepared old ultrastructural resources that lasted several months [1, 13, 14].

Cloud service providers need to reassure customers that their needs will be fulfilled completely [15–17].

1.1. Motivations and Goals. Using virtualization-based networks and storing bits for cloud computing can be a way to balance the load on networks; just as in the phase map and radius created in the big data, it can upset the balance. Using virtualization methods and load balancing methods can help a lot. In this paper, we present an algorithm PSGO considering scheduling. The main challenge about earlier researches during past years was that although resource scheduling and request management were taken into place, there was not considerable attention for energy optimization. In this research, we considered energy optimization as the main objective and tried to present a solution based on this. The energy-performance tradeoff can be considered for a physical machine that has computing ability in its peak usage. For optimizing energy consumption, we used the PSO optimization algorithm [14]. In this approach, user applications are mapping to virtual machines through the autonomous domain layer. Therefore, resource scheduling has been done with the assistance of an autonomous domain and virtual machine layer. The advantage of this heuristic algorithm is that it starts searching from some areas of space and extends the exploration domain and develops a stronger algorithm. In this work, we do the following to improve results: the final result of the PSO is to initiate population for GA. Thus, the first population of the GA is the optimal solution of the last step (Figure 1). Bestowing to the goal of this study is reducing load power in the cloud network based on load balancing, and the fitness function measures the load balance between the cloud network (host) servers.

In this study, we have dealt with the issue of virtualized reserve controlling of cloud computing records middles and, in particular, the load balancing among the servers, and we have done this by considering and implementing particle swarm optimization algorithms and genetic procedures [3, 18]. The sequel displays that the combination of these two can be effective in reducing the cost of resource allocation and load balancing. On the other hand, considering the combination of two optimization algorithms, the program execution time is increased compared to the implementation of an optimization algorithm. In this study, to improve the results, we will do the following: the final result of the PSO algorithm will be the initial value used in the genetic algorithm. Therefore, the initial population used in the genetic algorithm is the optimal solution obtained from the previous step. This method is very suitable for answering the main question of the current research due to its ability to convert the load balancing problem to the optimization problem (reduction of the imbalance cost function) and has been used in studies in this field as well.

In the first part of this article, some of the definitions related to this field and the introduction are discussed. The second section discusses the research background. In the third part, LBPSGORA based on the heuristic algorithms is presented [19], and, in the fourth part, an implementation of the proposed method and the result of the power consumption reduction are presented. In Section 5, we present future work and conclusions.

2. Literature Review

The load balancing algorithms are in three categories: (1) sender initiated load balancing algorithms, (2) receiver initialed load balancing algorithms, and (3) symmetric load balancing algorithms, a combination of both modes. The Round Robin algorithm is a load balancing method with an optimization energy consumption methodology. This issues tasks to all secondary processors. All tasks are assigned to secondary processors based on Round Robin (RR-Order) (Figure 2). Resources are selected with the machine to run in the alone direction. Selecting processors locally and independently of the allocation of processors is performed on each processor. The RR-Order algorithm generally is recycled in web servers where generally the WEB applications have the same nature and, as a result, they are evenly distributed. This defined pattern has not been under the principle of SLA [20].

With increasing the number of intelligent communication monitoring system devices such as WSN, WBANs, Big Data, SDN, and mCloud (mobile-Clouds), these devices have become moveable, proficient, and personal accounting infrastructure for the end handler in various modes [2, 8, 17, 21–25]. These devices come with cloud network intermediaries and IoT and SDNs, which cause them to enter the global Internet resources [12, 17]. Many end-users tend to use these devices instead of desktops or clients. In this case, CloudIoT and fog computing have appeared as a practical explanation, and their tasks can be offloaded on the cloud substructure [1]. As another example, network

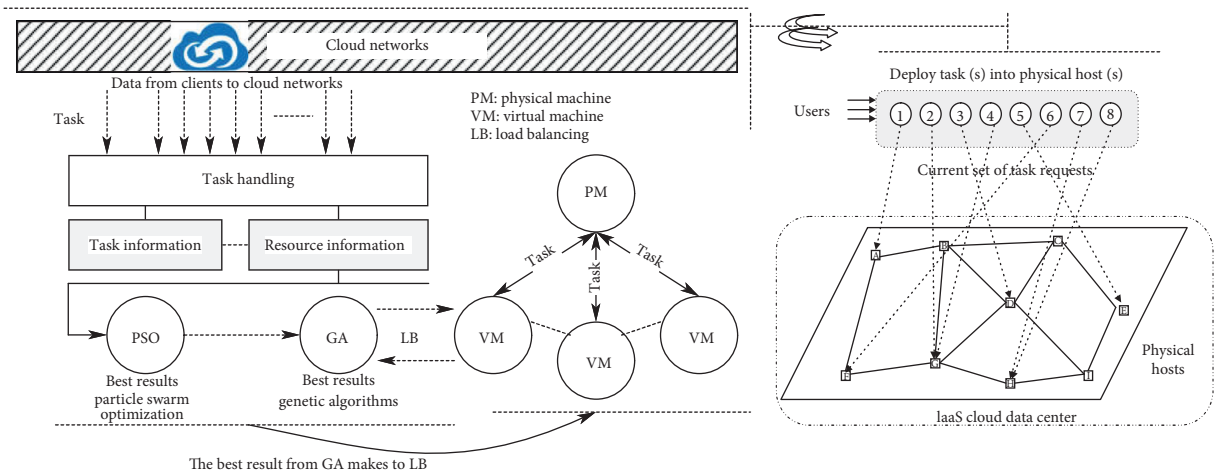

Figure 1: The framework of an algorithm based on load balancing to optimize energy consumption.

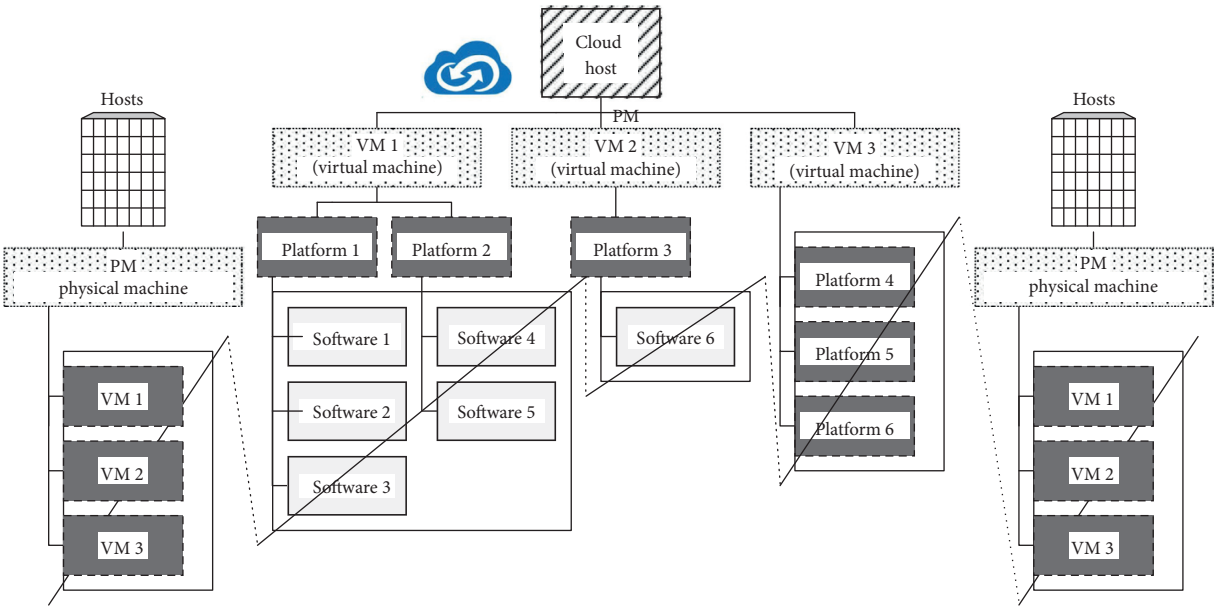

Figure 2: The diagram of cloud base structure.

bandwidth may change over time, and the performance levels of data rate cannot be guaranteed for task scheduling [18]. This is especially regarded in the area of real-time applications, where minimum latency is important. Resource management allocated to the mobile cloud must be considered in the cloud infrastructure. Because load-free applications and tasks run on the cloud substructure, fog resources such as storing are recycled to complement mobile device home-grown resources [9, 26].

In this regard [2], power consumption in CDC (CloudDataCenters) has been used. In this paper, the goal is to achieve an efficient resource management system to reduce operating costs and ensure service quality. The model of reducing energy consumption in this paper is based on continuous control of virtual machines using efficient methods. The main tool is the use of live migration. In this case, they can transfer virtual machines to physical machines without any interruption and with the highest flexibility, so that they can use the highest capacity of physical machines.

In [16], genetic algorithms for allocating virtual machines to reduce energy consumption in data centers are proposed, which shows that evolutionary algorithms are useful for allocating virtual machines because they have been able to reduce energy consumption by up to 10%.

In [27], a method of load balancing using PSO was provided. The authors have used many mobile cloud solutions to conserve many properties resources such as power and energy in mobile-portable devices. They optimized the grid energy by examining the SLA and placing the virtual machines using the PSO method.

In [28], in PSOSA, the PSO is used to minimize the energy. In experiments using cellular communications ad hoc mobile, the load balancer decision-maker used cellular convergence by PSO by load balancing on the network. The authors tried to evaluate their mechanism on a cloud-based cellular network.

Another method of load balancing the MOPSO multi-objective code transfer interval is proposed: the Energy Management and Service Quality Awareness (QOS) Framework [29]. The authors have used the Internet of Things for their inputs. They have considered the infrastructure for the network. Their approach focuses on portable mob-cloud service transfer created on QOS as well as power constrictions. If using the mobile device remote service is costly or the latency time is so high that QoS is not available, the fog service can be transmitted by communication and locally at a lower cost. It proposes algorithms for migration and transfers alongside the decision process for fog-grown performance against inaccessible performance.

In [30], a solution was presented based on the migration of virtual machines using PSO. The authors have used mobile devices to deliver cloud-based services. As part of their method, they supported fuzzy to apply multiple services; for example, a service output can be used as another service input. Their goal is to reduce energy consumption in the network, and resources and the potential of this platform in data are shared by online services. They have analyzed this potential and mentioned the difficulty of this way.

In [31], a C-RAN application parsing solution was presented; it works by splitting Apps to different mechanisms; then these modules are activated. The authors also discussed how to avoid the middleware solution from allocating further mob-resources to the fog server by using the OFDMA and PSO facilities instead of allocating all virtual machines. In the results section, several experimentations were achieved with m-devices to reach a DSS model to place the load on the network and reduce energy consumption.

In [32], a method with PSO considers reducing the consumption of available fog resources without the need for additional overflow in the m-device. In the obtained method, the implementation of PSO algorithm has been used to find suitable places in the network. Cloudlets and their modifications to find the suitable virtual machine location are another method for mobile clouds which are used like VM by placing load on the cloud infrastructure that may be nearby.

In [33], a scheduling method called NP-Hard is presented for the cloud. The authors have used PSO to reduce costs. How the cloudlet approach to scaling is used according to their resources, how m-devices network efficiency will cooperate with multiple users, and just how fog cloud will manage the stimulus ability need to be explored. The authors have introduced the CSOS method and optimized network energy by reducing makespan. In their method, the calculated cost has a low value.

In [34], the analysis of energy consumption by PSO on mobile devices has been examined to some extent in articles and writings. As mentioned, most of power and costs connected with using an m-device come from "DPRA" energy, known as the network interface. When the connection is made, the net edge remains active for each outstanding packet from the fog server, and it is costly for net operation. The work done in this paper is to implement a system called VM-DPRA, which groups network needs and requests together to reduce the number of tail energy occurrences. Their tail-ender method similarly attempts to cluster network requirements to minimize the power. In this research, K-means clustering method has been used for mapping tasks and microgenetic algorithm for dynamic aggregation of virtual machines. It has also been able to reduce the execution time.

In [28], a dynamics algorithm is presented. This algorithm is proposed to create load balance in cloud computing environments using the ant colony algorithm. This algorithm is one of the biological algorithms and almost simulates the behavior of ants.

The authors in [29] provide a method for allocating virtual machines using a linear search to optimize resources. The architecture used in this paper is that data centers include several physical machines and physical machines have a fixed size of RAM, CPU, and memory. The broker is responsible for managing user's requests; then the requests are transferred to the data center via the cloud, where the center manager reserves a certain number of virtual machines to execute the users' request and provides different classes of virtual machines by cloud maker is suggested, and the user can choose from these classes that have special features such as RAM, CPU, and memory. In this way, it has been able to reduce energy consumption to some extent.

The issue of proper placement of virtual machines in physical machines has a great impact on optimizing energy consumption. In [30], the ant colony system has been used to place virtual machines. In this research, the dynamic placement system of virtual machines has been considered. In this way, the request for several virtual machines changes in each time interval; at the beginning of the time interval, the list of virtual machines needs to be updated, and this happens by deleting the expired virtual machines before the previous interrupt. Then the resources of these expired machines are released, and new virtual machines are given to them. Updated virtual machines receive the right information and resources, which improves energy efficiency.

The study in [7] develops new VM scheduling to reduce energy costs for cloud services providers. In this study, it is assumed that there is an optimal frequency for one PM for VM processing. The optimal rate is also defined for the performance-power ratio based on which the heterogeneous PMs are weighted in the cloud environment. This problem is defined for the time-bound networks and has achieved good results. The workflow process is divided into several periods, where in each one the virtual machines have been dedicated to the proper PM and each core is operating at its optimal frequency. After each period, the cloud reconfigures to consolidate computing resources, which further reduces energy consumption.

Table 1 illustrates some related works with metaheuristics algorithms for solving time scheduling problems and improving energy consumptions in cloud networks.

3. The Proposed Method (LBPSGORA)

3.1. Problem Statement. Here, $ADS = AD_1, AD_2, \ldots, AD_r$ represents the autonomous domains set and each AD represents an autonomous domain. Each autonomous

domain has several virtual machines, as shown in Figure 3 [18, 35]. The cloud is mapped user applications to the virtual machine by running user applications and through a layer of autonomous domains. Therefore, the source environment scheduling of the cloud should be done by the layer of autonomous domains and the layer of virtual machines.

So here we have used two layers for scheduling; in this way, first, we can find autonomous domains that have more fitness and then search for the best virtual machines for assigning tasks among virtual machines at those domains. In addition, we considered some autonomous tasks as an application and attempted to assign them to the virtual machines by the proposed scheduling method. Thus, a hierarchical scheduling framework is used for the proposed algorithm. To illustrate this problem, we present a two-dimensional form in Figure 4, in which X_{axis} shows speed of CPUs and Y_{axis} shows the amount of RAM (Table 2 and Table 3).

3.2. Proposed Method LBPSGORA. In this study, a load balancing algorithm is presented for optimizing energy consumption by considering scheduling. The problem with research that has been done over the last years is that, despite addressing resource allocation and requests management, there has not been much attention paid to energy efficiency in it. But this research aims to provide a solution based on this vital problem, considering energy efficiency as the primary goal. The energy efficiency ratio for a physical machine (PM) that contains the computational power ratio at the peak of energy can be used for this purpose. The algorithm used to optimize energy consumption is the particle swarm optimization algorithm. In this method, users' applications are mapped to virtual machines through a layer of autonomous domains. Therefore, the scheduling of cloud environment resources is done by the layer of autonomous domains and the layer of virtual machines. The advantages of using metaheuristic algorithms are that one can start searching from several areas of the problem space and increase the level of exploration and have a stronger algorithm. In this study, to improve the results, the final achieved result of the PSO algorithm will be used as an initial value in the genetic algorithm. Therefore, the initial population used in the genetic algorithm yields is the optimal response obtained from the previous step.

3.3. Particle Swarm Optimization Algorithm. In this part, we describe the several specifics of the LBPSGORA. As virtual machines are the subset of physical machines that are processing tasks, the PSO algorithm can work in a complex problem to find suitable tasks. The formulas that are presented in this section are related to the particle swarm optimization method. This method is based on using metaheuristic algorithms, in such a way that the metaheuristic methods have been used for assigning VM to PMs. The PSO algorithm is a global optimization method that can solve problems with solutions in a point or surface in n-dimensional space. Therefore, in addition to the initial velocity of particles, the channels of communication between the particles are identified. By moving these particles into the solution space, the results are evaluated based on a merit criterion. A large number of swarm particles make the method flexible against the local optimal solution problem. Each particle has a location. $x_i(t)$ specifies the position of particle i at time t. Each particle also needs a swiftness to move through space. $v_i(t)$ specifies the velocity of particle (i) at time (t). The equations for updating the particle positions are given below. The particles are talented at memorizing the best position in their lives. The best individual experience of a particle or the best position met by a particle is called y_i. The particles know the best position met in the whole group. This is called the $\widehat{y}$ position. The two values of c_1 and c_2 are constant and w is the initial weight that is decreased linearly from $w_{\max}$ to $w_{\min}$. $t_{\max}$ is the maximum number of repetitions intended for algorithm (Table 4). Therefore, it can be expected that, by utilizing metaheuristics in the initial allocation of VM to PMs, it would be closer to the optimal solution and reduce the number of migrations (Figure 5, part A).

$$
\begin{aligned}
v_i^{t+1} &= w * v_i^t + c_1 * \text{rand}_1 * \left(y_i - x_i^t\right) + c_2 * \text{rand}_2 * \left(\widehat{y} - x_i^t\right),\\
x_i^{t+1} &= x_i^t + v_i^{t+1},\\
w &= w_{\max} - t * \frac{w_{\max} - w_{\min}}{t_{\max}},\\
v_{\min} &\le v_i^t \le v_{\max}, \quad i = 1, \ldots, n,\\
x_{\min} &\le x_i^t < x_{\max}, \quad i = 1, \ldots, n.
\end{aligned}
\tag{1}
$$

3.4. Genetic Algorithm. The steps used in the genetic algorithm are as follows: at first, building the initial population is based on the best solution obtained from the PSO algorithm. Selecting and combining initial samples (arrays) to generate a new sample or a crossover are done in this algorithm. In the crossover phase, a random combination of two fathers was used to generate a new child. If the generated children contain repeated genes, they will be replaced by the number of unused genes. The whole crossover phase is performed in ten iterations with a combination of 10 randomly selected pairs. Modifying existent samples and building a new sample prevent local optimization because similar samples may be used for the combination in the crossover phase. Selecting a sample with the best fitness among the produced samples in this generation continues to produce the next generation of samples until the end condition is met (Figure 5, part B).

On the path to the aim and on the way to achieving the optimal solution for the problem, it is necessary at each step to measure the proximity of the current solution to the aim using a criterion. This is done by the fitness function. In this paper, the fitness function is carried out by using the study in [17]. To achieve an optimal solution where there is load balancing between cloud (hosts) network servers, we need to minimize the mean deviation between the loads allocated to

Table 1: Comparison of load balancing methods for cloud networks.

Reference	Description	Method	Advantages	Disadvantages	Simulations
[27]	Load balancing for optimization problem	Using modified PSO for load balancing	To maximize user-centric energy efficiency in the uplink communication	High capacity, low throughput, time complexity	MATLAB
[28]	Cloud provider load balancing PSOSA	Using PSO for having high performance LB in fog	VMs is presented to decrease LB on fog cloud	Has a multiplex physical resource and high time complexity	CloudSim
[29]	Load balancing with placement optimization MOPSO	PSO algorithm for time scheduling and improving energy	Low energy consumption	High delay and low throughput, time complexity	MATLAB
[30]	Load balancing decentralized particle swarm optimization	PSO-GA-based is presented to decrease LB on CloudNets	Low energy consumption, solve the optimization problem of maximizing the profit	High total execution time	MATLAB
[31]	Task scheduling problem on cloud is a load balancing for clouds with load balancing C-RAN	Scheduling constitutes one of the crucial aspects of resources management system in cloud computing	Improve energy consumption and resource utilization	High total execution time (makespan), bad throughput, and low quality of service	CloudSim
[32]	Cloud task scheduling problem with the aim of decreasing power	PSO-based for minimization energy and LB	Low energy consumption	Have a scheduling problem between virtual machines	CloudSim
[33]	CSOS energy-saving research of virtualization of the cloud computing platform with load balancing	PSO algorithm for solving time scheduling	Reduces energy with migration and high quality of service	Time complexity, high delay	MATLAB
[34]	DPSRA resource allocation VM-DPRA	PSO algorithm for solving time scheduling	Low energy consumption	Time complexity, high delay	MATLAB
[18]	Hybrid load balancing task scheduling problem PSO and GA	Novel load balancing symbiotic organisms search PSO + GA	Low cost, solving task scheduling	Time complexity and low quality of service	CloudSim

the servers. Suppose that T_i denotes all computing powers of a server and L_i is the computational power left by the server after assigning it to requests. The computing power of a server is calculated by the speed of CPU and the number of storage resources and its memory.

If the number of servers or hosts is m (the m physical hosts are heterogeneous),

$$\begin{aligned} T_i &= \alpha T_c^i + \beta T_r^i, \\ L_i &= \alpha L_c^i + \beta L_r^i, \\ i &= 1, 2, \ \ldots, m. \end{aligned} \tag{2}$$

Therefore, according to the standard deviation formula to optimize the load balancing in the cloud network, we use the following fitness function, where $E(X)$ is a mathematical expectation (Figure 6):

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(X_i - E(X)\right)^2}. \tag{3}$$

The remaining load ratio in a server is defined as (residual load rate)

$$E_i = \frac{L_i}{T_i}, \quad i\epsilon\{1.2.3, \ldots, m\}. \tag{4}$$

Therefore, the fitness function is defined as the standard deviation function:

$$\sigma = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(\frac{\alpha L_c^i + \beta L_r^i}{\alpha T_c^i + \beta T_r^i} - \frac{\sum_{k=1}^{m}\left(\alpha L_c^k + \beta L_r^k\right)/\left(\alpha T_c^k + \beta T_r^k\right)}{m}\right)^2}. \tag{5}$$

4. Simulation and Results

In the cloud computing environment, some slaves are the central interfaces between users and the cloud infrastructure. This slave is responsible for resources management policy and allocating requests for machines. In addition to the slave that manages the cloud with its policies, due to the distributed nature of the cloud computing network, distributed resource management algorithms can also be provided. The purpose of this research is to manage resources in the cloud environment and to present a new method for examining the current status and history of semicentralized cloud-level machines. The cloud environment includes the physical infrastructure and specific control functions that the cloud, with their collaboration, will be able to provide and manage services. These infrastructures and resources are integrated and configured by the infrastructure provider in the data centers. The cloud provider

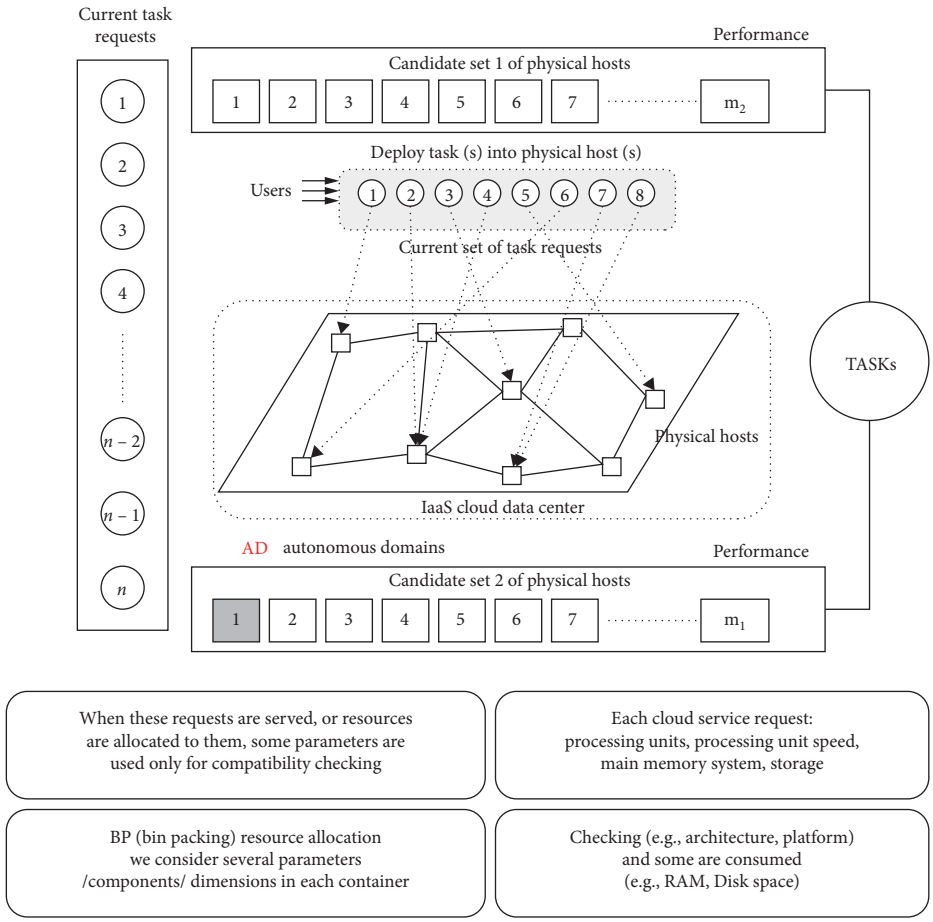

Figure 3: The application scheduling framework on the cloud.

provides resources to developers with the help of virtualization techniques and resource management algorithms (Figure 7). We used MATLAB and CloudSim to evaluate the algorithm. The simulations have been run in an environment with an Intel processor and 6 GB of RAM. The operating system used is 64-bit Windows 7 Ultimate (Tables 5 and 6).

MATLAB's programming language is used to simulate the peer-to-peer system, connect between records, and construct the data distribution. Figure 8 illustrates the implementation steps of this research. After performing this initialization, Figure 9 shows the number of resources and registered requests in the program when the LBPSGORA is evaluating.

4.1. Applying PSO to Data Resources Matrix. In the process of implementing LBPSGORA after identifying the resource matrix and also the submissions matrix, the PSO_Algorithm has been applied to data. So we can allocate sent requests appropriately so that the load balancing between the servers is maintained and the cost function defined for this research will be minimized. Table 7 shows the parameters used in the particle swarm optimization algorithm.

After the implementation of the PSO algorithm according to the initialization, the manner of initial and final allocation to some of the requests is given in the following table as an example. Comparing Table 8 and Table 9, it can be seen that the final cost considered to the resource allocation is much lower than the initial cost. Figure 10 shows a comparison of the initial cost and the final cost in resource allocation to the requests for a better understanding of how the algorithm works. Also, Figure 11 shows the amount of resource allocation cost per replicate. As can be seen, this algorithm can find the optimal path correctly and after 100 iterations can propose the desired results at the lowest cost. Table 7 and Table 8 represent the details of how to allocate a resource to received requests.

The number of resources is considered to be 100, where, for each of these resources, the parameters related to CPU capacity, memory, and so forth are given values according to the initial values in Table 9.

4.2. Applying Genetic Algorithm. After applying the PSO to data related to the defined requests and resources in the system, it is time to implement the genetic algorithm. Thus, after performing the PSO and achieving the best cost in allocating resources for the received requests to balance genetic servers, the GA_algorithm is applied to the best results of the particle swarm optimization algorithm. Thus, we believe that performing this method can reduce the cost of resource allocation for received requests.

In this research, we also intend to minimize the cost defined. By comparing Table 10 with the final output of the

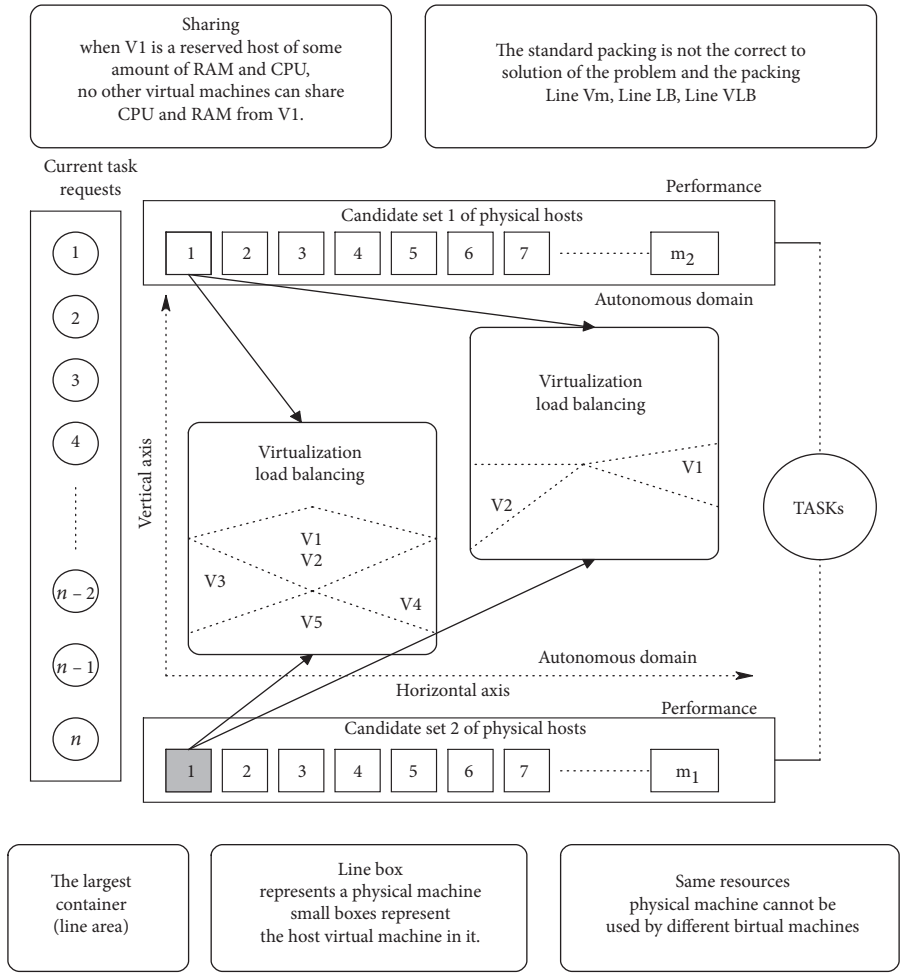

Figure 4: A hierarchical scheduling framework is used for AD.

Table 2: An example of available resources.

Application number	CPU (GHz)	RAM (GB)	HDD (T)
1.1	7.88	32	2.
2.1	7.44	32	2.
3.1	6.48	16	2.
4.1	6.48	16	2.

Table 3: Example of requests received by the cloud.

Application number	CPU (GHz)	RAM (GB)	Memory (GB)
1.1	3.03	12	100
2.1	2.04	12	200
3.1	3.18	32	200
4.1	2.15	32	200
1.1	4.12	32	400
2.1	4.24	8	100
3.1	7.22	16	400

PSO algorithm and Table 11 showing the parameters of the genetic algorithm, it can be concluded that the final cost considered to allocate resources and load balancing has been reduced desirably in comparison to the cost obtained in the PSO algorithm. Therefore, the LBPSGORA can propose appropriate allocations that maintain the load balancing between the servers. Figure 12 shows a comparison of the initial cost and the final cost in allocating resources for the requests for a better understanding of how the algorithm works. Also, this method reduces the cost of resources to requests, and Figure 12 shows the amount of allocation costs of resources in each iteration. This algorithm can find an optimal path and can propose the desired results with the minimum cost after 100 iterations.

4.3. Runtime Calculation. In this sector, we investigate the runtime calculation for the LBPSGORA, PSO algorithm, and genetic algorithm. To be fair to the obtained results, we assume that the numbers of rings are equal, as depicted in Figure 13. The numerical value of the LBPSGORA runtime in different repetitions is shown in Table 12.

The variable of the cost of consumption is taken as a threshold to load balancing in the cloud network and optimizing energy consumption. By spending more time and developing a hybrid algorithm, we have improved energy consumption and cost. Without load balancing, when clients send requests to the cloud network, data transmission operations are complicated, and retransmission occurs. As a result, it needs to spend money and extra energy (Table 12).

Table 4: Resource parameters and their initialization range.

Components	Application	Description
$\mathbf{v}$	The velocity of each particle	Each particle has a velocity vector such that $v_i = (v_{i1}, v_{i2}, \ldots, v_{in})$
$\mathbf{x}$	The location of each particle	Each particle has a position vector such that $x_i = (x_{i1}, x_{i2}, \ldots, x_{in})$
$\mathbf{C}_1, \mathbf{C}_2$	Two random coefficients	c_1 , c_2 are acceleration coefficients. The acceleration of each particle is directed to $\mathbf{P}_{\text{best}}$ and $\mathbf{G}_{\text{best}}$
$\mathbf{p}_{\text{best}}$	Best value	The particle's best known position
$\mathbf{g}_{\text{best}}$	The best cost considered	Moving to the best position that has been in the neighborhood of the swarm from the beginning until now

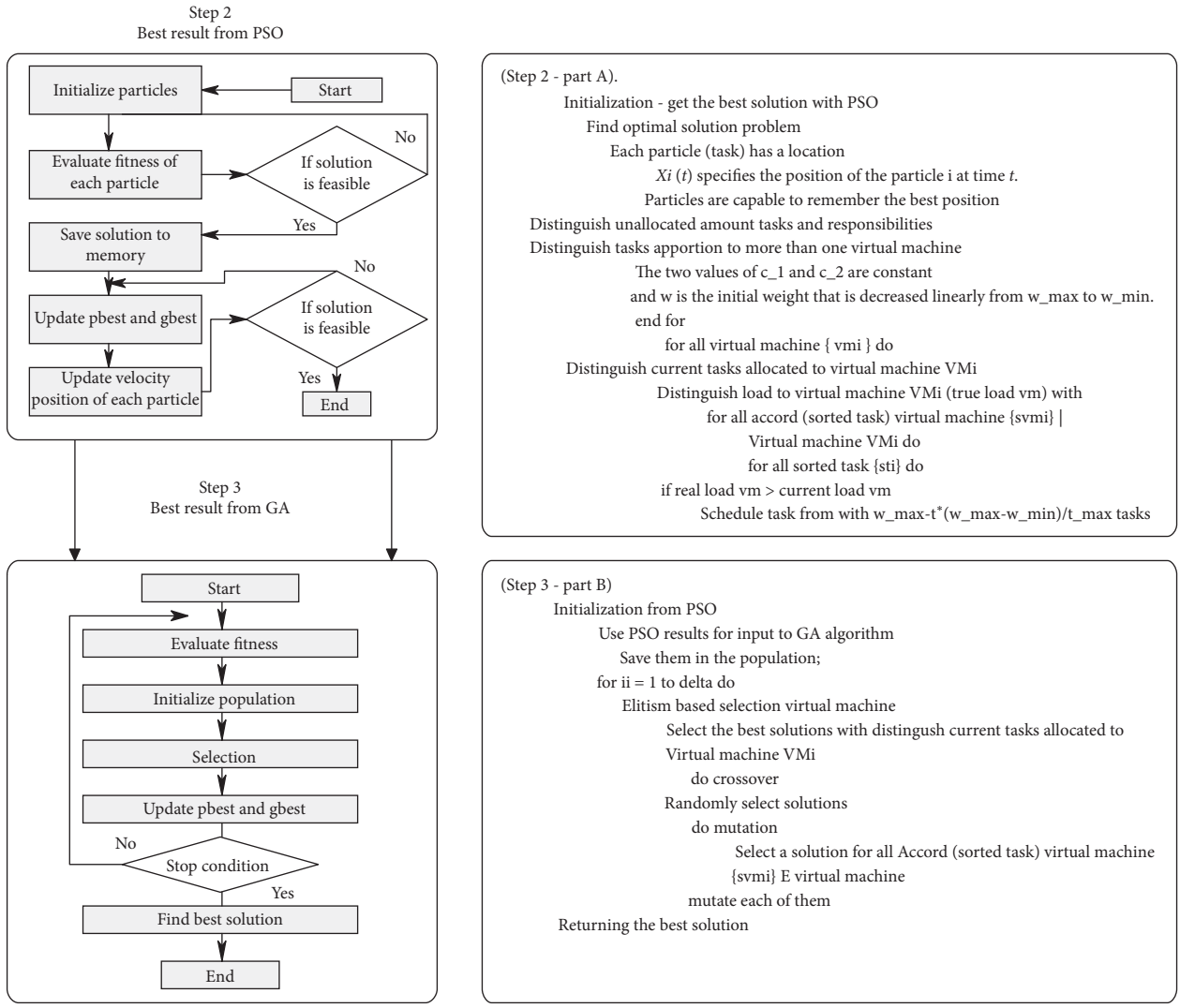

Figure 5: The proposed algorithm. Step 2 is related to PSO for distinguishing tasks apportion to more than one virtual machine and step 3 is related to GA for returning the best solution.

4.4. Check the Load Balance against Tasks. In this section, the details of LB in exchange for increasing tasks in the physical machine are discussed. In the proposed method, the LBPSGORA combination is intended to handle tasks.

The proposed method is compared with the method in [16], hybrid GA-PSO, and GA algorithms. In hybrid GA-PSO method, due to having higher complexity compared to our proposed method, it has a weaker load balancing compared to our proposed method. The difference between our work and the article is that the authors therein used a combination of PSO + GA for their method and the two methods depend on internal steps to create the initial population. In LBPSGORA method, two separate algorithms work, and the optimal PSO response for the initial population of GA is considered.

As can be seen in Figure 14, the proposed method has better conditions for increasing tasks. This makes it less

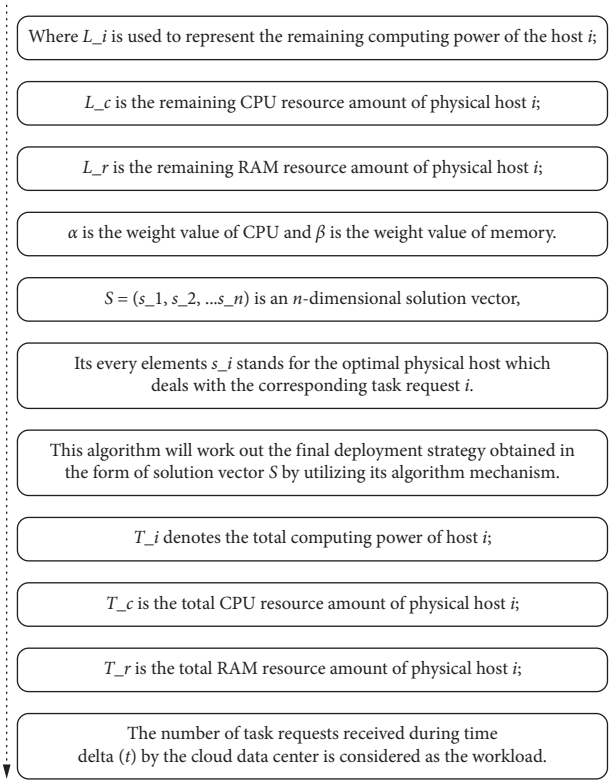

Figure 6: Each step to measure the proximity of the current solution to the aim using a criterion.

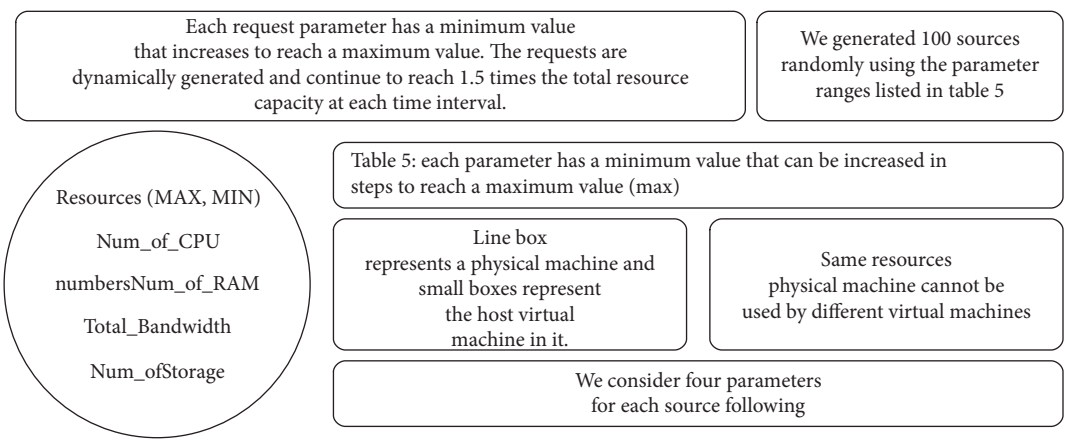

Figure 7: Resources management policy and allocating requests.

Table 5: Resources and initialization range.

Resources (MAX, MIN)	Num_of_CPU numbers	Num_of_RAM	Total_Bandwidth	Num_of Storage
MinRe	8 (2.4 GHz)	32.2	10	1
MaxRe	7 (3.8 GHz)	32.4	8	200
StepsRe	4 (2.5 GHz)	32.7	4	200

Table 6: An example of requests and their range.

Resources (MAX, MIN)	Num_of_CPU numbers	Num_of_RAM (GB)	Total_Bandwidth (GB)	Num_of Storage (TB)
Maximum	7.00 (4.2 GHz)	16	10	1
Minimum	11.00 (4.4 GHz)	256	1	100
Steps	5.00 (7.2 GHz)	256	1	100

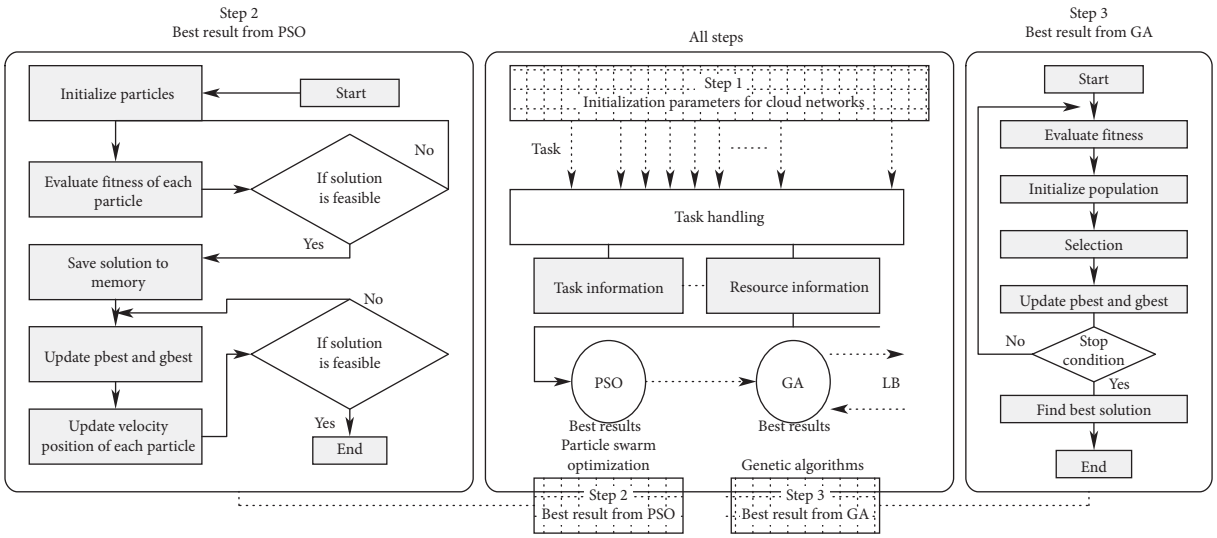

Figure 8: The steps of this research (LBPSGORA algorithm).

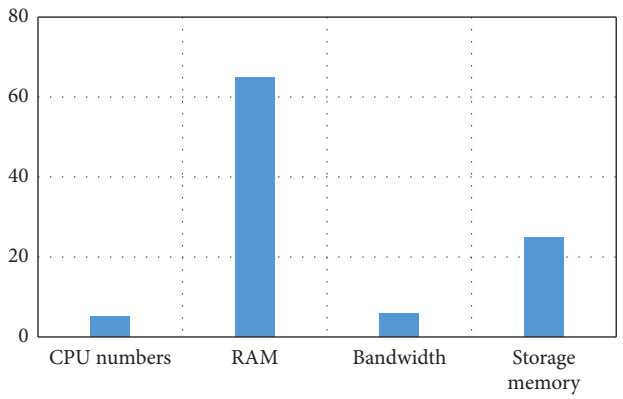

Figure 9: The average number of each virtual machine in the resource parameters.

Table 7: The parameters used in the PSO algorithm.

Parameter	Description	Value
nVar	Number of decision parameters	1
MaxIt	Maximum number of rings	1000
W	Initial weight	1
Wdamp	Weight change coefficient	0.99
c1	Personal learning rate	1.5
c2	Basic learning rates	2

Table 8: Initial values attributed to requests from 1 to 3.

Request ID	Resource ID assigned	Request value in parameter order	The number of resources in the parameter order	Cost
1	88	[3, 11, 6, 2]	[6, 64, 8, 35]	0.0618
2	94	[3, 35, 5, 9]	[3, 96, 10, 21]	0.0617
3	17	[2, 51, 6, 3]	[7, 128, 8, 7]	0.06204

Table 9: Final values attributed to requests from 1 to 3.

Request ID	Resource ID assigned	Request value in parameter order	Source value in parameter order	Cost
1	35	[3, 11, 6, 2]	[8, 112, 10, 23]	0.0615
2	26	[3, 35, 5, 9]	[12, 141, 5, 10]	0.0616
3	45	[2, 51, 6, 3]	[25, 150, 20, 20]	0.0616

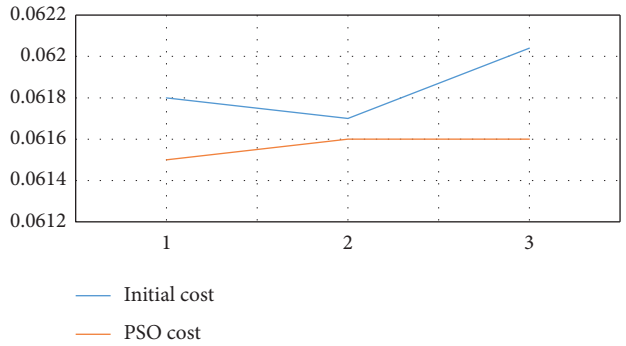

Figure 10: Comparison of the initial cost and final cost of assigning the source to requests from 1 to 3.

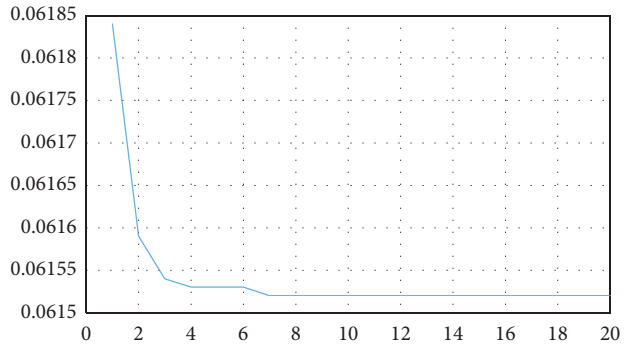

Figure 11: The cost of resource allocation in different replays of implementation of the PSO algorithm.

Table 10: Parameters used in the genetic algorithm.

Parameter	Description	Value
nPop	Population size (number of request parameters)	4
n_gen	Number of generations	1000
Rp	Crossover percentage or crossover	0.8
Mu	Percentage of mutation or mutation	0.5

Table 11: Final values attributed to requests from 1 to 3.

Request ID	Resource ID assigned	Request value in parameter order	The amount of resources in the parameter order	Cost
1	35	[3, 11, 6, 2]	[25, 308, 52, 71]	0.0491
2	26	[3, 35, 5, 9]	[28, 303, 58, 64]	0.0608
3	45	[2, 51, 6, 3]	[25, 332, 51, 78]	0.0600

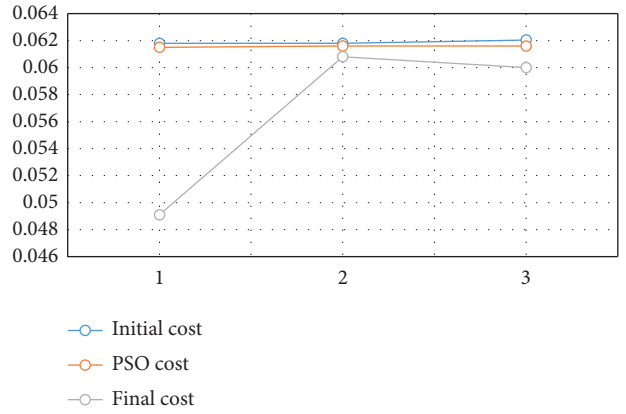

Figure 12: Comparison of the initial cost and final cost in resource allocation to requests 1 to 3.

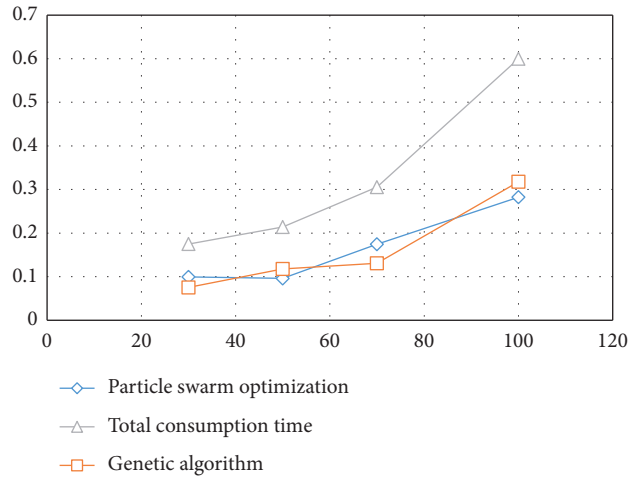

Figure 13: Comparison of the runtime of the LBPSGORA in different repetitions in seconds.

Table 12: The numerical value of the LBPSGORA runtime in different repetitions.

Number of iterations	PSO algorithm/t (s)	GA algorithm/ t (s)	Total time spent/t (s)
30	0.099411	0.075396	0.174807
50	0.096196	0.117715	0.213911
70	0.174365	0.130853	0.305218
100	0.282268	0.317631	0.599899

energy-consuming. The PSO and GA methods are also unfavorable conditions for observing the LB of the other methods. The circles in Figure 15 show the effect of increasing tasks on LB. With the increase of tasks, we see that the load created on the network has increased. As shown in Figure 13, the proposed method is similar to the compared methods in different cases and the improvement rate of the proposed method compared to hybrid GA-PSO is 8.42%, and it is 10.61% compared to GA and 11.71% compared to PSO.

Also, the simulation results for tasks 100 to 1000 are examined in detail in Figure 15. The improvement rate of LBPSGORA method compared to hybrid GA-PSO, respectively, is as follows: 5.12% for 100 tasks, 3.67% for 200 tasks, 4.18% for 400 tasks, 2.28% for 600 tasks, 5.12% for 800 tasks, and 3.23% for 1000 tasks.

4.5. Check Makespan against Task Execution. This section compares the proposed LBPSGORA method with the hybrid GA-PSO method and the number of makespan changes against the number of tasks. As shown in Figure 16, as the number of tasks increases in different cases, the proposed method and other methods will be changed. Because the tasks in the proposed method are more efficient than the hybrid GA-PSO method, makespan changes in various tests are due to the use of two-stage PSO and GA and network updates. In some incremental modes, load balancing changes are also observed. The proposed method is similar to other compared methods in different cases and the improvement rate of the proposed method is 7.32%.

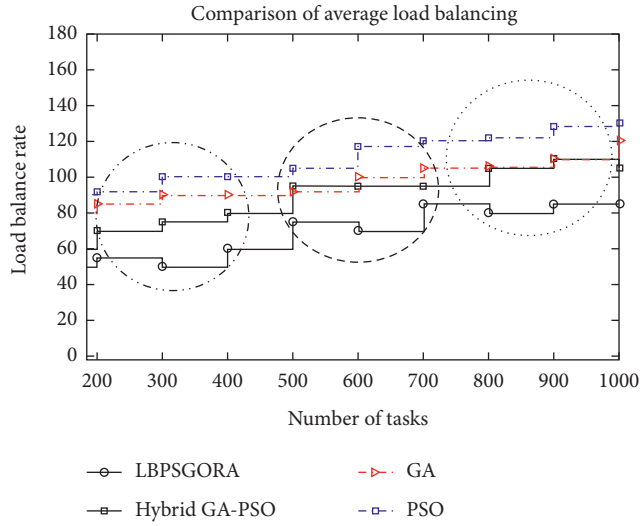

Figure 14: Load balancing rates in LBPSGORA and hybrid GA-PSO.

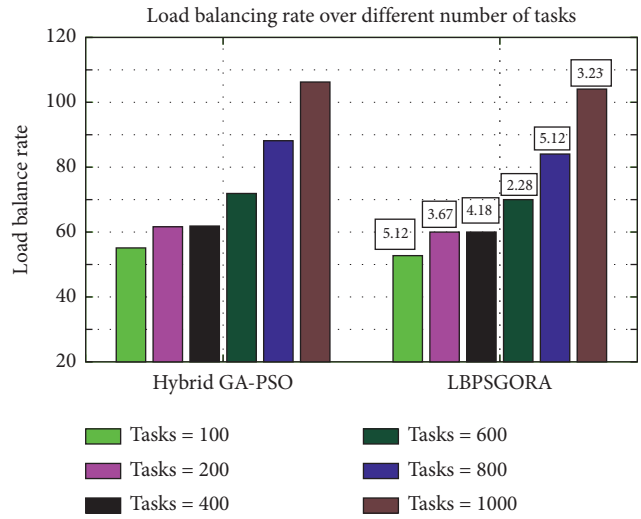

Figure 15: Load balancing changes in LBPSGORA and hybrid GA-PSO.

Also, the simulation results for the 100 to 1000 tasks are discussed in detail in Figure 15. The improvement rate of LBPSGORA method compared to hybrid GA-PSO in makespan, respectively, is as follows: 2.1% for 100 tasks, 1.39% for 200 tasks, 1.36% for 400 tasks, 2.74% for 600 tasks, 1.06% for 800 tasks, and 2.60% for 1000 tasks.

4.6. Task Execution Cost. Figure 17 examines the costs of task execution changes in different implementation of tasks, and, as can be seen, the proposed method has a lower implementation cost task execution compared to other methods. In the hybrid GA-PSO method, the cost changes are similar to the performance of the proposed method and the LBPSGORA is 6.87% more efficient. As can be seen, with the increase in the number of tasks, the cost task execution for both methods of implementation costs is increasing. Also, the simulation results for 100 to 1000 tasks are reviewed in detail. The

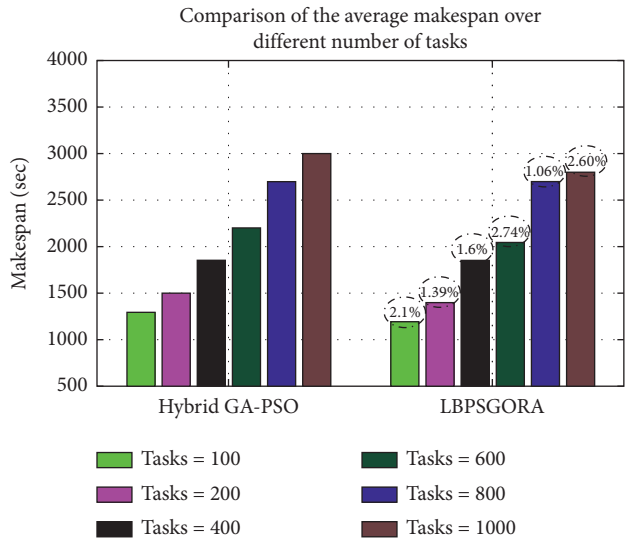

Figure 16: Makespan implementation changes in LBPSGORA and hybrid GA-PSO.

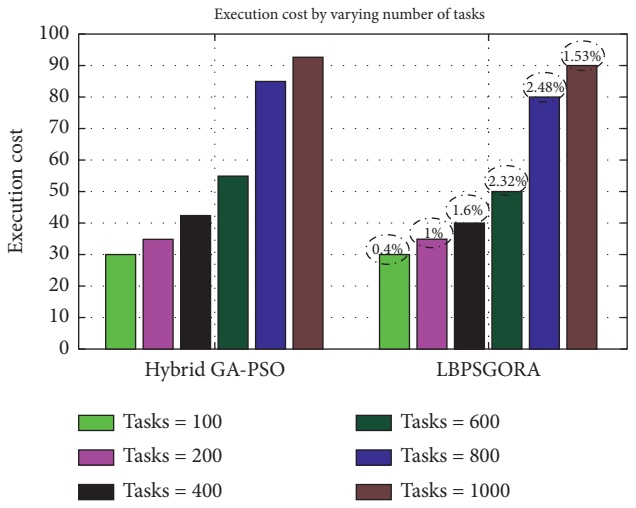

Figure 17: Execution cost in the LBPSGORA and hybrid GA-PSO.

improvement cost task execution rate of LBPSGORA compared to hybrid GA-PSO, respectively, is as follows: 0.4% for 100 tasks, 1% for 200 tasks, 1.6% for 400 tasks, 2.32% for 600 tasks, 2.48% for 800 tasks, and 1.53% for 1000 tasks.

5. Conclusions

The most significant challenges that cloud service providers face are managing physical resources and scheduling tasks by taking into consideration the reduction of energy consumption throughout the network. In the LBPSGORA approach, instead of randomly assigning the initial population to the genetic algorithm, the best result obtained from the PSO algorithm is considered as the initial population in GA. The hybrid GA-PSO method is similar in performance to the proposed method with tasks changes. The LBPSGORA method is 6.87% more efficient in execution cost and 7.32% more efficient in makespan compared to hybrid GA-PSO. In load balancing, LBPSGORA was 8.42% superior to the hybrid GA-PSO, 10.61% superior to GA, and 11.71% superior to PSO. For future work of this research, the use of other heuristic methods with load balancing and scheduling problems will be considered. Besides, reliability and response time considerations are other areas of future research.

Data Availability

The data underlying this article will be shared on reasonable request to the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] S. M. Mirmohseni, C. Tang, and A. Javadpour, "Using markov learning utilization model for resource allocation in cloud of thing network," *Wireless Personal Communcations.*vol. 115, pp. 653–677, 2020.

[2] A. Javadpour, G. Wang, S. Rezaei, and K.-C. Li, "Detecting straggler MapReduce tasks in big data processing infrastructure by neural network," *Journal of Super Computing*, vol. 76, pp. 6969–6993, 2020.

[3] M. Farid, R. Latip, M. Hussin, and N. A. W. Abdul Hamid, "A survey on QoS requirements based on particle swarm optimization scheduling techniques for workflow scheduling in cloud computing," *Symmetry (Basel)*, vol. 12, no. 4, 2020.

[4] S. M. Bozorgi, M. R. Hajiabadi, A. A. R. Hosseinabadi, and A. K. Sangaiah, "Clustering based on whale optimization algorithm for IoT over wireless nodes," *Soft Computing*, vol. 25, no. 7, pp. 5663–5682, 2021.

[5] A. Mosa and N. W. Paton, "Optimizing virtual machine placement for energy and SLA in clouds using utility functions," *Journal of Cloud Computing*, vol. 5, no. 1, p. 17, 2016.

[6] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers," *Concurrency and Computation: Practice and Experience*, vol. 24, no. 13, pp. 1397–1420, 2012.

[7] Y. Ding, X. Qin, L. Liu, and T. Wang, "Energy efficient scheduling of virtual machines in cloud with deadline constraint," *Future Generation Computer Systems*, vol. 50, pp. 62–74, 2015.

[8] A. Javadpour, G. Wang, S. Rezaei, and S. Chend, "Power curtailment in cloud environment utilising load balancing machine allocation," in *Proceedings of the 2018 IEEE SmartWorld, Ubiquitous Intelligence Computing, Advanced Trusted Computing, Scalable Computing Communications, Cloud Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, pp. 1364–1370, IEEE, Guangzhou, China, October 2018.

[9] A. K. Sangaiah, M. Y. Suraki, M. Sadeghilalimi, S. M. Bozorgi, A. A. R. Hosseinabadi, and J. Wang, "A new meta-heuristic algorithm for solving the flexible dynamic job-shop problem with parallel machines," *Symmetry (Basel)*, vol. 11, no. 2, 2019.

[10] A. K. Sangaiah, S. R. Ali, S. R. H. Ali et al., "Energy-aware geographic routing for real time workforce monitoring in industrial informatics," in *Proceedings of the IEEE Internet Things Journal*, p. 1, NewYork, NY, USA, February 2021.

[11] S. G. Domanal and G. R. M. Reddy, "Optimal load balancing in cloud computing by efficient utilization of virtual machines," in *Proceedings of the Communication Systems and Networks (COMSNETS), 2014 Sixth International*, pp. 1–4, Bangalore, India, January 2014.

[12] A. Javadpour, "Providing a way to create balance between reliability and delays in SDN networks by using the appropriate placement of controllers," *Wireless Personal Communications*, vol. 110, pp. 1057–1071, 2019.

[13] A. Javadpour, G. Wang, and S. Rezaei, "Resource management in a peer to peer cloud network for IoT," *Wireless Personal Communications*, vol. 115, pp. 2471–2488, 2020.

[14] A. Javadpour, N. Adelpour, G. Wang, and T. Peng, "Combing fuzzy clustering and PSO algorithms to optimize energy consumption in WSN networks," in *Proceedings of the 2018 IEEE SmartWorld, Ubiquitous Intelligence Computing Advanced Trusted Computing Scalable computing. Communications Cloud Big Data Computing Internet People Smart City Innovations*, pp. 1371–1377, Guangzhou, China, October 2018.

[15] A. Priyadharshini and T. Nadu, "A survey on security issues and countermeasures in cloud computing storage and a tour towards multi-clouds," *International Journal of Engineering and Technology*, vol. 1, no. 2, pp. 1–10, 2013.

[16] A. Javadpour, K. Saedifar, G. Wang, and K.-C. Li, "Optimal execution strategy for large orders in big data: order type using Q-learning considerations," *Wireless Personal Communications*, vol. 112, pp. 123–148, 2020.

[17] A. Javadpour, "Improving resources management in network virtualization by utilizing a software-based network," *Wireless Personal Communications*, vol. 106, no. 2, pp. 505–519, 2019.

[18] A. M. Manasrah and H. Ba Ali, "Workflow scheduling using hybrid GA-PSO algorithm in cloud computing," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 1934784, 16 pages, 2018.

[19] J. Zhao, K. Yang, X. Wei, Y. Ding, L. Hu, and G. Xu, "A heuristic clustering-based task deployment approach for load balancing using bayes theorem in cloud environment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 2, pp. 305–316, 2016.

[20] S. Banerjee, M. Adhikari, and U. Biswas, "Development of a smart job allocation model for a Cloud Service Provider," in *Proceedings of the Business and Information Management (ICBIM), 2014 2nd International*, pp. 114–119, Durgapur, India, January 2014.

[21] A. Javadpour, "An optimize-aware target tracking method combining MAC layer and active nodes in wireless sensor networks," *Wireless Personal Communications*, vol. 108, pp. 711–728, 2019.

[22] A. Javadpour and H. Memarzadeh-Tehran, "A wearable medical sensor for provisional healthcare," in *Proceedings of the ISPTS 2015 - 2nd International. Symposium on Physics.and Technologyof Sensors Dive Deep Into Sensors*, pp. 293–296, Pune; India, March 2015.

[23] A. Javadpour, H. Memarzadeh-Tehran, and F. Saghafi, "A temperature monitoring system incorporating an array of precision wireless thermometers," in *Proceedings of the International Conference on Smart Sensors and Application (ICSSA)*, pp. 155–160, Kuala Lumpur, Malaysia, May 2015.

[24] A. Javadpour, G. Wang, and X. Xing, "Managing heterogeneous substrate resources by mapping and visualization based on software-defined network," in *Proceedings of the IEEE International Conference on Parallel Distributed Processing with Applications, Ubiquitous Computing Communications, Big Data Cloud Computing, Social Computing Networking, Sustainable Computing Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*, pp. 316–321, December 2018.

[25] F. Ja'fari, S. Mostafavi, K. Mizanian, and E. Jafari, "An intelligent botnet blocking approach in software defined networks using honeypots," *Journal of Ambient Intelligence Humanized Computing*, vol. 12, no. 2, pp. 2993–3016, 2021.

[26] A. K. Sangaiah, M. Sadeghilalimi, A. A. R. Hosseinabadi, and W. Zhang, "Energy consumption in point-coverage wireless sensor networks via bat algorithm," *IEEE Access*, vol. 7, pp. 180258–180269, 2019.

[27] P. Jain and S. K. Sharma, "A Systematic Review of Nature inspired Load Balancing Algorithm in Heterogeneous Cloud Computing Environment," in *Proceedings of the 2017 Conference on Information and Communication Technology (CICT)*, pp. 1–7, Ghaziabad, India, November 2017.

[28] B. Huang, W. Wei, Y. Zhang et al., "A task assignment algorithm based on particle swarm optimization and simulated annealing in Ad-hoc mobile cloud," in *Proceedings of the 2017 9th International Conference on Wireless Communications and Signal Processing (WCSP)*, pp. 1–6, October 2017.

[29] T. Kumrai, K. Ota, M. Dong, J. Kishigami, and D. K. Sung, "Multiobjective optimization in cloud brokering systems for connected internet of Things," *IEEE Internet of Things Journal*, vol. 4, no. 2, pp. 404–413, 2017.

[30] F. Ramezani, M. Naderpour, and J. Lu, "A multi-objective optimization model for virtual machine mapping in cloud data centres,," in *Proceedings of the IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), 2016*, pp. 1259–1265, Vancouver, BC, Canada, July 2016.

[31] Y. Sun, C. Li, Y. Huang, and L. Yang, "Energy-efficient resource allocation in C-RAN with fronthaul rate constraints," in *Proceedings of the 2016 8th International Conference on Wireless Communications Signal Processing (WCSP)*, pp. 1–6, Yangzhou, China, October 2016.

[32] S. Wang, Z. Liu, Z. Zheng, Q. Sun, and F. Yang, "Particle swarm optimization for energy-aware virtual machine placement optimization in virtualized data centers," in *Proceedings of the 2013 International Conference on Parallel and Distributed Systems*, pp. 102–109, Seoul, Korea, December 2013.

[33] M. Abdullahi, M. A. Ngadi, and S. I. Dishing, "Chaotic symbiotic organisms search for task scheduling optimization on cloud computing environment," in *Proceedings of the2017 6th ICT International Student Project Conference (ICT-ISPC)*, pp. 1–4, Johor, Malaysia, May 2017.

[34] L.-D. Chou, H.-F. Chen, F.-H. Tseng, H.-C. Chao, and Y.-J. Chang, "DPRA: dynamic power-saving resource allocation for cloud data center using particle swarm optimization," *IEEE Systems Journal*, vol. 12, no. 2, pp. 1554–1565, 2018.

[35] D. Pandit, S. Chattopadhyay, M. Chattopadhyay, and N. Chaki, "Resource allocation in cloud using simulated annealing," in *Proceedings of the Applications and Innovations in Mobile Computing (AIMoC), 2014*, pp. 21–27, Kolkata, India, February 2014.