Research Article

iMedPub Journals
www.imedpub.com

American Journal of Computer Science and Information Technology
ISSN 2349-3917

2020
Vol. 9 No. 1: 68

Presentation of a Multi-Agent Method to Improve Load Balance and Provisioning Dynamic Resources in the Cloud Computing Environment

Meghdad Hoseinpour*

Department of IT Engineering, Dezful Branch,Islamic Azad University, Dezful, Iran

Abstract

Background: Cloud computing is an emerging technology that provides multiple processing and storage services over the Internet. Users can store data on the cloud instead of storing it on their devices, thus provide comprehensive access to the data, and also they can executed their applications on powerful cloud computing platforms, so there is no need to install software. Challenges: In such environments, providing resources has become a challenging issue. Another challenge in cloud computing is balancing the load between computing nodes. In resent researches, scheduling or balancing challenges have usually been addressed of one aspect. So that most systems have a factor that is either responsible for scheduling or used for balancing, and in limited multi-factor systems there is little use of new or meta-heuristic algorithms.

Method and Finding: Accordingly, in this paper, by presenting a two-factor system based on two meta-heuristic algorithms, particle swarm optimization and ant colony optimization, an attempt is made to balance the load on the cloud infrastructure as well as reduce task execution time as well as energy consumption.

Conclusion: According to the comparison made and CloudSim simulation output, the proposed method is more efficient than cat swarming methods as well as the single-factor system.

Keywords: Load balance improvement; Dynamic resource supply; Cloud computing

Received: November 12, 2020, **Accepted:** November 26, 2020, **Published:** December 02, 2020

***Corresponding authors:**
Meghdad Hoseinpour

Department of IT Engineering, Dezful Branch, Islamic Azad University, Dezful, Iran

Hoseinpourmeghdad@gmail.com

Citation: Hoseinpour M (2020) Presentation of a Multi-Agent Method to Improve Load Balance and Provisioning Dynamic Resources in the Cloud Computing Environment. Am J Compt Sci Inform Technol. Vol. 9 No. 1: 68.

Introduction

Cloud computing is an emerging technology that provides various processing and storage services over the Internet. Cloud service providers rent data center software and hardware to provide their storage and processing services over the Internet. Using cloud computing, Internet users can execute their applications on powerful cloud computing platforms and thus there is no need to install software [1]. In such environments, sourcing has become a challenging issue, because it needs to avoid supplying less than needed so that the program does not slow down and it also needs to avoid oversupply of resources so as not to lead to unnecessary resource costs [2]. Another challenge in cloud computing is to balance the load between computing nodes. Load balancing increases efficiency, reduces response time and optimizes resource utilization. This action distributes the load between the nodes in the system, causing the load to be uniform between the nodes and prevents nodes from being idle and overloading other nodes. So it can be said that time, cost and balance is three important parameters to solve the problem of scheduling requests sent by the user [3]. In 2014, Singh et al. developed an effective scheduling and load balancing algorithm for public and private clouds in cloud computing. They have developed an algorithm for the private cloud that uses SJF with starvation and also considers the load balancing problem. The result of load balancing in the private cloud shows that the less loaded virtual machine is selected to execute the user request, which in turn increases private cloud output for public load [4]. And colleagues in 2015 proposed a load balancing algorithm based on the autonomy factor in cloud computing. This research focuses on load balancing in the cloud computing environment. The general distribution of this mechanism is the calculation of the virtual machine proactive load in a data center and whenever the load of a virtual machine approaches the threshold value, the load agent starts searching for a candidate virtual machine from other data centers. Retaining information from the candidate virtual machine reduces service time. The result obtained through implementation proves that this algorithm works satisfactorily [5]. In 2015 introduced a virtual machine migration technique for load balancing. A set of front-rear operating load balancing

experiments to determine the initial hosts of virtual machines and a distributed problem-solving technique to distribute the load evenly among the hosts. The results show that agent participation can efficiently balance loads in a high-performance distributed method compared to business solutions. This business solution called Red hat, which includes the rate of virtual machine migration. However, this is a focused approach that is not fully scalable and still suffers from a high virtual machine migration overflow [6]. In comprehensive study on the implementation of two loads balancing algorithms called Max-Min and Min-Min based on their chosen cloud environment. The evaluation results in this paper show that Max-Min performs better than Min-Min in terms of work time. There are other studies of load balancing in the cloud that show that the Min-Min algorithm works better than the Max-Min. An algorithm is superior to the other one in terms of cloud environment, so both algorithms have their pros and cons. If the number of lighter tasks is greater than the heavier tasks, Max-Min then has complete superiority over Min-Min in terms of source usage and execution time; conversely, if heavier tasks are more, Min-Min works better than Min-Max. So we can conclude that the load balancing performance in the cloud does not depend on any algorithm; rather, it is entirely based on the cloud environment we choose [7]. In load balancing mechanism based on a maintenance policy for distributing of requests across duplicate servers. This method allows overloaded servers to store the capacity of remote servers before sending requests to them. The simulation results show that the proposed technique offers better response times; fewer discarded requests, better resistance to faulty load changes, better balance and less message overflow [8]. In developed a load balancing and resource management framework for swift to bridge bottlenecks and optimize resource utilization [9]. This template is a distributed and widely used storage system on the cloud. The proposed template is lightweight and does not require any modification of the source code of the guest operating system and storage system. In addition, the proposed template regulates system performance, improves resource usage calculation and also ensures the reliability of the storage system by using the virtual machine migration method as its migration method. However, using a low resource causes high costs for hardware resources and cannot always improve system performance [10]. Inspired by bee load balancing in the search behavior of bees for load balancing in virtual machines. Virtual machine load calculation, load scheduling and balancing decision, virtual machine grouping and task scheduling. Test results show improvements in QOS delivered to customers, minimization of virtual machine migrations, and execution time. However, it still suffers from low scalability [11]. With the maximum study done in this regard, using a newer and more efficient multi-factor system to improve load balance and resource allocation in the cloud computing environment, required to reduce the execution time and reduce costs. Our main goal in this research is to optimize the works and providing the program by shortening the completion time for the work of the customers, while minimizing the related costs. In this study, we consider the cloud computing environment to be dynamic. We are looking to design a system that has a multi-factor system and several restrictions, such as restrictions on the resources of cloud service providers; Consider customer satisfaction and power consumption [12]. The proposed system should monitor the resources of cloud service providers such as CPU power consumption, memory, permanent storage, network bandwidth, and so on. To improve resource management, optimal decisions are made based on the characteristics of the virtual machine, to which customer demands are assigned to each of them depending on the SLA. Virtual machines should be selected in a way that has the maximum effect on overloading. In general, virtual machine displacement occurs when the host physical machine encounters with a problem of overlap or slow loading. In this case, we will use the virtual machine relocation algorithm.

Methods

A two-factor system is used to create optimal scheduling as well as load balancing; one of these factors is responsible for scheduling and allocating resources with the help of particle swarm optimization algorithm and the other factor is responsible for balancing the load by using the ant colony algorithm [13]. In the research, scheduling or balancing challenges have usually been addressed from one dimension. So that most systems have a factor that is either responsible for scheduling or used for balancing, and in the limited multi-factor systems have not benefited much from new algorithms or meta-exploration. Accordingly, in this paper, by presenting a two-factor system based on two meta-heuristic algorithms, particle swarm optimization and ant colony optimization, attempts are made to balance the load on the cloud infrastructure, as well as reduce task execution time as well as energy consumption **(Figure 1)**.

As shown in **Figure 1**, in the first step, the ready to execute tasks are scheduled on the processing servers using the particle swarm optimization algorithm. In this step, first, the amount of free resources of the servers is calculated through parameters that are added to the PSO algorithm and tasks are sent to candidate servers for execution, accordingly. Then, after scheduling and starting the processing of tasks, using the ant colony algorithm, these servers are periodically examined in the next phase. In order to determine if they have experienced heavy traffic or congestion due to an error (hardware or software), their tasks will be transferred to other candidate servers using the migration technique. Using this two-factor system, in addition to reducing the execution time of tasks, also it is possible to increase the reliability of the cloud infrastructure by creating a kind of optimal load balance As a result, the groundwork for improving the service

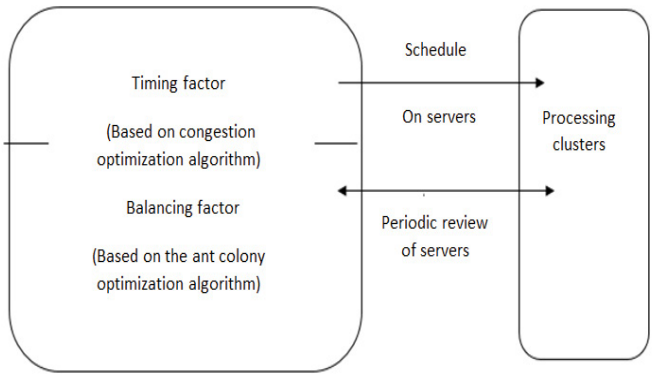

Figure 1: Overview of the proposed mode.

level agreement, which is one of the important parameters to ensure the quality and satisfaction of the user, should be provided. Accordingly, in the continuation of this chapter, these two sub-factors are fully described.

Ant-based balancing agent

Before applying ant colony optimization (ACO) algorithms as a balancing factor, the problem under study should be defined as a graph G = (C, L) in which C is a finite set of components and L ⊆C × C is the set of connections between components. The optimal solutions to the problem can be expressed as possible paths in graph G. Therefore, ACO algorithms can be used to find optimal paths with minimal cost. In ACO algorithms, a population (colony) of agents (ants) collectively solves the problem under study [14]. The information collected by the ant during the search process is encoded according to the search path between two points (Lij) in the pheromone sequence (τij). Pheromone sequences encode a long-term memory of the complete ant search process and are updated by the same ants. The value of the initiative, often called innovation information, actually represents the previous information about the sample problem or execution time information. In many cases, the original η information, the estimated cost of adding new components or connections to the answer, is being generated. These values are used by the ants' innovative laws in possible decisions to move [15].

It should be noted that ants act simultaneously and independently and although the function of each ant is complex enough to find an answer to the problem, the best answer a high quality answer will only emerge through group and group interactions among ants. These interactions are mediated by indirect communication, by reading and writing information that ants use in variables that store pheromone values.

Initialization of pheromones

In cloud infrastructure, the resources allocated to nodes and virtual machines are not the same and usually change dynamically. Due to this feature, in this paper, according to the method presented in, the physical resources of virtual machines are used to determine the initial amount of pheromone related to nodes. The following three physical sources are used to quantify the pheromone;

- Processor (number of cores and number of executable instructions per core) (p)
- Storage space (m)
- Bandwidth (b)

The defined capacity for the physical resources of virtual machines is also defined in below **(Table 1)**.

Accordingly, the pheromone quantification for a function node can be defined as Equation (2):

$$\tau_i = \psi_1 \tau_{band}(0) + \psi_2 \tau_{Mem}(0) + \psi_3 \tau_{GPU}(0)$$

In this regard, ψ_n is a weight factor and is used to regulate the power of physical resources in the cloud computing infrastructure.

Pheromone update

In this study, the main purpose of pheromone updating is to increase the amount of pheromone in the processing nodes executing the virtual machine, which has the necessary conditions for load balancing, and node reduction is a function that does not meet the requirements [16]. In the proposed solution, the following two influential factors are considered to update the amount of pheromone.

Table 1: Capacities of physical resources.

Capacities of physical resources	
Processor	$\tau_{cpu}(0) = \frac{p_{cpu}}{p_{\max}} \times 100$
Storage space	$\tau_{mem}(0) = \frac{m_i}{m_{\max}} \times 100$
Bandwidth	$\tau_{band}(0) = \frac{p_b}{p_{b\max}} \times 100$

- Pheromone evaporation time
- Adding new task

Particle swarm-based scheduling agent

In this section, the scheduling factor is an imitation-based method to create an optimal scheduling of the PSO algorithm. In this algorithm, the population is equal to the number of particles in the problem space. The particles are randomly initialized. Each particle will have a compatibility value and will be evaluated by a compatibility function that must be optimized in each generation [17]. In other words, this solution is equivalent to a bird in the pattern of collective movement of birds. Each particle has a merit value calculated by a merit function. The closer the particle is to the target in the search space (food in the bird movement model), the greater the merit is. Each particle also has a velocity that directs the particle's motion. Each particle continues to move in the problem space by following the optimal particles in the current state.

Suppose xi (t) denotes the position of the particle i in the search space in time step t; the position of the particle vi (t), changes to the current position, by adding speed. In other words, we have

$$x_i(t+1) = x_i(t) + v_i(t+1)$$

Schedule tasks using particle swarm optimization

In this solution, the whole swarm is considered as the neighbor of each particle. The social component of particle velocity updating is obtained according to all the information obtained from the swarm particles. In this method, social information includes the best places found by the crowd which is called ŷ (t). Accordingly, the velocity of particle i is calculated as relation.

$$v_{ij}(t+1) = v_{ij}(t) + c_1 r_{1j}(t)[y_{ij}(t) - x_{ij}(t)] + c_2 r_{2j}(t)[\hat{y}_{ij}(t) - x_{ij}(t)]$$

In the above relation v_{ij} (t) the velocity of particle i is in the range j=1,...nx and in time step t.x_{ij} (t) is the position or location of the particle i in the time step t and C1 and C2 are positive constants called acceleration coefficients that used to regulate the degree of influence of social and cognitive components.

r_{1j} (t) and r_{2j} (t) are random values in the range [1 and 0] that are sampled from a uniform distribution.

In order to create an optimal scheduler by PSO algorithm, first the average cost of all tasks calculated on all processing resources. This cost can be calculated for all programs by performing each task on a specific set of resources. This cost is calculated and stored in a matrix called TP. As processing costs are inversely related to processing time, the cost is higher for resources that complete the task faster. Similarly, the average amount of transmission costs between resources is stored in units of data, as represented by the PP matrix. It should be noted that the transfer cost has an inverse ratio to the time used. It is also assumed that the input and output sizes of each task are specified. In addition, the transfer fee is calculated in seconds. The first step is to calculate the mapping of all tasks in the workflow regardless of their dependencies. This mapping optimizes the overall cost of calculating the workflow of the program. To validate the dependencies between tasks, the algorithm assigns ready -made tasks to resources, based on the mapping provided by the particle swarm algorithm. Ready-to-use tasks are tasks that have been completed by their father and provided the necessary files to perform the tasks. After submitting tasks to resources for execution, the scheduler waits for polling time. This is the time to get the status of tasks and resources. Now, according to the number of completed tasks and also determining the amount of energy consumption and resources, the list is ready to be updated. That this time will include tasks that have been completed by their father. The average values for the transfer between sources are then updated based on the current network load. The transcripts must be recalculated when the transfer fee changes. Also, when remote resource management systems are unable to outsource the task to resources based on the desired mappings due to unavailability, recalculating the PSO creates a dynamic heuristic balance of task mappings. The finished tasks are assigned to the processing resources based on the recalculation of the maps. These steps continue until all tasks in the workflow are scheduled. This algorithm is dynamic as the transfer cost in each schedule loop is updated based on the average transfer time between resources. Also, considering that the task-resource mapping is recalculated in this solution, so the processing cost is optimized based on the current network and resource conditions.

To calculate the amount of network bandwidth consumption and the amount of traffic generated by each physical machine, Equation (5) is used:

$$D_j = \sum_{\forall m \in Vj} \lambda(\mathrm{j},\mathrm{m}) \sum_{i=1}^{\rho(\mathrm{j},\mathrm{m})} C_i$$

D_j: The degree of j machine physical communications with other physical machines.

$\lambda(j\vartheta m)$: The traffic load between the j physical machine and the m physical machine.

V_j: The set of physical machines that are associated with the j physical machine.

C_i: The weight of the link between two physical machines at level i.

$\rho(j\vartheta m)$: The level of relationship between m physical machines and j physical machine.

According to the above defined parameter, the cost function for modeling network traffic in the cloud computing environment is as a relation:

Minimize $\sum_{j=1}^{m} D_j$

$$\sum_{j=1}^{m} X_{ij} = 1 \quad \forall i \in I \tag{1}$$

$$\sum_{i=1}^{n} R_{D_i} . X_{ij} \le T_{D_j} . \mathrm{Y}_j \quad \forall j \in j \tag{2}$$

$\mathrm{Y_j}\vartheta\mathrm{X_{ij}} \quad \forall \mathrm{j} \in \mathrm{J}$ and $\forall \mathrm{i} \in \mathrm{I}$

i: A collection of virtual machines.

j: A set of physical machines.

R_{Di}: The amount of bandwidth requested for each virtual machine.

T_{Di}: The high bandwidth consumption threshold in each physical machine.

X: is a binary variable that if virtual machine 'i' is assigned to physical machine j, it is one otherwise it will be zero.

Y_j: is a binary variable whose value is one if the physical machine j is in use, otherwise it is zero.

Condition (1) maps the virtual machine i to only one physical machine. Condition (2) models the capacity of physical machines, and finally condition (3) expresses the range of problem variables.

Implementation and evaluation

Cloud sim is developed in Java and can be used to simulate IaaS, SaaS and PaaS clouds. The scalability of this simulator is very high and it can simulate very large clouds. Currently, in addition to simulating a cloud, cloud sim is able to simulate very large cloud environments consisting of several clouds.

In order to measure the solutions and create a workload, a load manufacturer is needed. In this study, actual survey data from the CoMon project, which is indeed a monitoring infrastructure for planet lab servers, is used. In this solution, the status of physical machines is periodically examined and their data are assumed to be at intervals of 300 seconds.

To examine the proposed method, we compare it with the following two load balancing solutions, which have already been done in cloud infrastructure:

- Cat Crowd Optimization Strategy (CSO): In this strategy, cats are placed in two searches and tracking modes, In fact, their two main behaviors are modeled with two sub-models called tracking and search mode. By combining these two relatively defined modes, the cat swarm optimization algorithm performs well.
- Ant Colony-Based Scheduling (ACO): The ant colony algorithm works on a set of answers called a population. The population of an ant colony is associated, and each ant in that population represents a solution or answer. Unlike genetic algorithms that new answers are created from existing answers in each step, in ant colony algorithms, the new answers are randomly selected at each step. In this solution, the ant is used instead of particles to find suitable

servers and schedule them. In this case, the processing servers are clustered and by sending the ant to a cluster, the server check operation is performed.

800 hosts are defined to simulate data centers. Also, in order to create any virtual server in Cloudsim software, it is necessary to create a class named Server that extends the Power Model Spec Power class, which is also from the Power Model class in the Cloudsim energy modeling package. And also, in the tests performed, the number of ants is considered equal to 100. In order to measure the proposed method in real time and with high load, the solution is tested based on real-time tracked data from the Co Mon project, which is a monitoring infrastructure for Planet Lab. This data can be downloaded from the github site.

In this data, CPU throughput collected by thousands of virtual machines from hosted servers around the world. The timing interval is assumed to be 300 seconds. Also, 10 days of randomly traversed data is selected. To compare the performance of the proposed algorithm with other solutions, the following criteria have been set for evaluation:

- Number of virtual machine migrations, since the migration technique is used to reduce the load on high end servers, this parameter indicates the number of migrations that has been done during the simulation process and to create balance.
- SLA degradation rate. Execution time, which is the amount of instructions execute per unit time.
- The amount of energy consumption is calculated in kilowatt hours (kwh) and indicates the total amount of energy spent on processing the desired data during the simulation process.

It should also be noted that all operations related to the production of workloads as well as their alignment in the list of ready or unfinished tasks, message propagation, alignment, bottlenecks, communication link properties, implement TCP / IP protocol, switching and transmission Packages as well as routing requests; simulated and performed by Cloudsim infrastructure. The evaluation and results of the output of the ClodSim simulator are shown in **Figures 2 to 5**.

Figure 2 shows the amount of energy consumption in kilowatt hours during the process of implementing the algorithms in the simulator environment. As it can be seen, the amount of energy consumption in the proposed solution has been significantly reduced compared to the two CSO and ACO solutions. This is due to the convenient scheduling created by using a scheduling factor based on particle swarm optimization. In addition, evenly

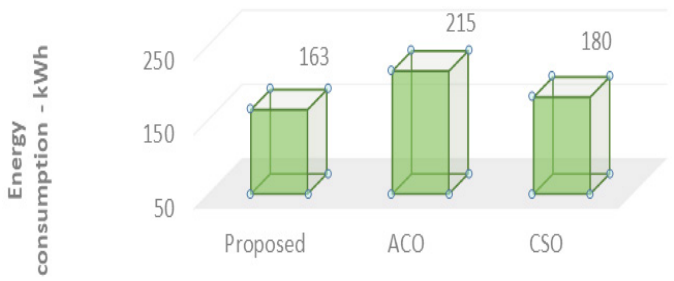

Figure 2: Energy consumption (KW/hr).

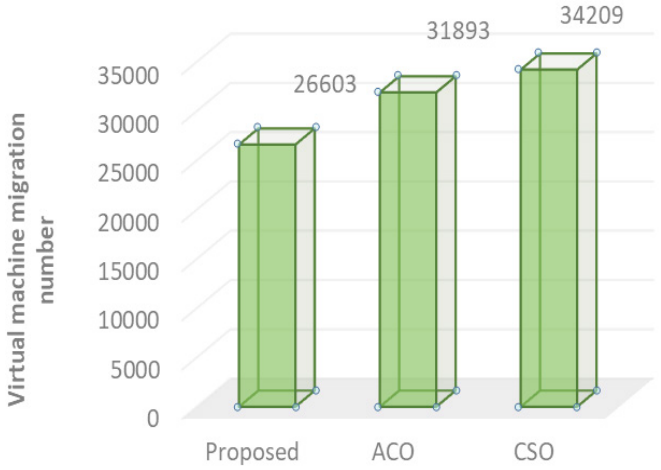

Figure 3: Virtual machine migration rate.

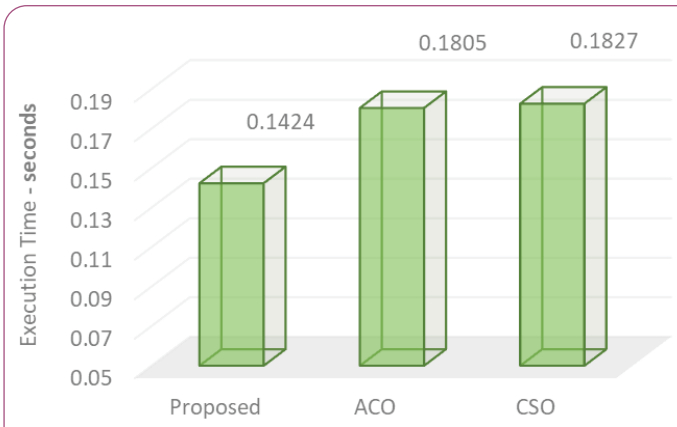

Figure 4: Execution time.

Figure 5: Percentage of violation of SLA (%SLA).

distributing the load between processing servers and shutting down idle servers through the balancing agent is another reason for this amount of optimization; there is a 24% reduction in energy consumption compared to cat swarm scheduling (CSO)

and about 10% reduction compared to PSO scheduling. This energy reduction was predictable, considering that the amount of resources and energy consumption is examined in each scheduling operation **(Figure 3)**.

Another goal of this research is to use the virtual machine migration technique in order to create the optimal load balance along with the balancing factor. In the case of virtual machine migration this point must be kept in mind that if migration is not optimal and appropriate, it can create a lot of overhead in the network and also use up server resources unnecessarily. As a result, the higher migration rate of a farmer cannot mean that it is better, as if the proposed migration method as shown in **Figure 3**, has had less migration than the proposed CSO and ACO solutions, but as will be seen below, the proposed solution, compared to the other two solutions, has been able to reduce the execution time as well as the amount of energy consumption, and even less is the error, therefore, as a result of the proposed method, although there were fewer migrations, but these migrations were appropriate and optimal. As a result, it has the least additional overhead on the network and does not unnecessarily interfere with server resources. In general, it can be concluded that the migration of the proposed solution has been done more efficiently and accurately, considering the use of energy consumption parameters and network traffic **(Figure 4)**.

In addition to reducing energy consumption, reducing application execution time is also of great importance in cloud infrastructure. In fact, this is another goal of this research, as shown in **Figure 4**, in the proposed solution, the execution time is reduced to a reasonable proportion, compared to CSO-based scheduling by 21% and compared to PSO by 22% reduction in execution time. The reason is to create a load balance in the cloud infrastructure which by using these two factors based on particle swarm optimization along with an ant colony-based load balancer using these two factors, overloading the servers is prevented, and as a result, tasks are processed in less time **(Figure 5)**.

Figure 5 shows the 5% violation of the Service Level Agreement (SLA) as another important output of the simulator.in fact; the service level agreement is the basis for determining the expected level of service. Service quality parameters in the service level contract specify how appropriate the quality provided was. Customers need this contract to ensure the quality of the services they need. Indeed, the main purpose of this contract is to define a formal basis for the terms of service provided, such as: efficiency, availability or issuance of a bill. As Figure 4, with the help of load balancing created of the optimal scheduler, the proposed solution has made the servers more reliable than the other two solutions. As a result, there is a higher availability of cloud infrastructure than other solutions, and in other words, compared to the other two solutions, the percentage of violation of this solution is less than the service level agreement.

Results

Cloud computing provides access to a wide range of virtual resources such as hardware, software and services that these services can serve their users dynamically and with a variable volume. Balancing and dynamic resource provision in cloud applications is a new challenge, to determine how much virtual resources should be allocated to each layer of application to minimize resource consumption and achieve service level agreement. General automation mechanisms and flexibility required. Accordingly, in this paper, in order to create optimal scheduling as well as load balancing, a two-factor solution based on particle swarm optimization and ant colony algorithm was used. The proposed technique prevents overload or under load on the servers by optimally assigning tasks to physical servers. In addition, through the balancing factor, which is based on the ant colony algorithm, it creates an optimal load balance to minimize the response time to the request, by distributing the tasks among the processing servers.

Finally, in a simulated environment, the proposed solution was compared with two equilibrium solutions based on cat swarm (CSO) and ant colony (ACO) and was shown to be more efficient than both solutions. And we have achieved a good goal of reducing execution time and energy consumption. Energy consumption has decreased by 24% compared to ant colony and more than 10% compared to CSO schedule. Execution time was also reduced by 21% compared to the ant colony-based schedule and by 22% compared to CSO which is due to the load balancing in the cloud infrastructure using the based schedule. Cats are crowded alongside the virtual machine migration technique, indeed, the use of virtual machine migration technique make it possible to transfer loads from busy servers to freer or no-load servers, resulting in a significantly reduced execution time. The degree of breach of the Service Level Agreement (SLA) is other parameter that has been examined. In fact, this agreement is the basis for determining the expected level of service. Service quality parameters in the service level contract specify how appropriate the quality provided was. In the tests performed, it was shown that the percentage of violation of this solution is less than the service level agreement compared to the other two solutions. In other words, the proposed solution has made the servers more reliable than the other two solutions, and as a result, higher availability has been created in the cloud infrastructure than other solutions.

Conclusion

This solution can also be used in cloud content distribution networks (CDNs). These networks duplicate and cache content from the main server to alternate servers spread across different geographical locations. In this way, the requested contents of the user are delivered from closer server. Substitute servers on a CDN only hold a specific set of content and serve requests for that set. Therefore, their rate of hit can reach one hundred percent. With the help of a cat-based optimization solution, the best content can be found and replicated on alternate servers. In fact, you can find the best content to cache and duplicate it on servers that can be found in search mode by generalizing this solution and using the tracking mode in cats. As a result, the efficiency of these cloud streams increases.

References

1 Qian L, Luo Z, Du Y, Guo L (2009) Cloud computing: An overview: In IEEE International conference on cloud computing 626-631.

2 Rittinghouse JW, Ransome JF (2016) Cloud computing: implementation, management, and security.

3 Kumar K, Lu YH (2010) Cloud computing for mobile users: Can offloading computation save energy. Computer 43: 51-56.

4 Kansal NJ, Chana I (2012) Cloud load balancing techniques: A step towards green computing. Int J Comp Sci Issu. 9: 238-246.

5 Armbrust M, Fox A, Griffith R, Joseph AD, Katz R, Konwinski A et al (2010) A view of cloud computing. Communications of the ACM 53: 50-58.

6 Chen H, Wang F, Helian N, Akanmu G (2013) User-priority guided Min-Min scheduling algorithm for load balancing in cloud computing. In national conference on parallel computing technologies 1-8.

7 Mell P, Grance T (2011) The NIST definition of cloud computing.

8 Kousalya K, Balasubramanie P (2009) To improve ant algorithm's grid scheduling using local search. Int J Comput Cogn 7: 47-57.

9 Tawfeek MA, El-Sisi A, Keshk AE, Torkey FA (2013) Cloud task scheduling based on ant colony optimization. In international conference on computer engineering & systems 64-69.

10 Sun J, Xiong SW, Guo FM (2004) A new pheromone updating strategy in ant colony optimization. In proceedings of International conference on machine learning and cybernetics 1: 620-625.

11 Velte T, Velte A, Elsenpeter R (2009) Cloud computing, a practical approach.

12 Khan S, Sharma N. Effective scheduling algorithm for load balancing (SALB) using ant colony optimization in cloud computing. Int J Adv Res Comput Sci Softw Eng 4: 966-973.

13 Lu X, Gu Z (2011) A load-adapative cloud resource scheduling model based on ant colony algorithm. In IEEE international conference on cloud computing and intelligence systems 296-300.

14 Mohan A, Shine S (2013) Survey on live VM migration techniques. Int J Adv Res Comput Eng Technol 2: 155-157.

15 Dasgupta K, Mandal B, Dutta P, Mandal JK, Dam S (2013) A genetic algorithm (GA) based load balancing strategy for cloud computing. Pro Technol 10: 340-347.

16 Liu H, Jin H, Xu CZ, Liao X (2013) Performance and energy modeling for live migration of virtual machines. Clus Compu 16: 249-264.

17 Ramezani F, Lu J, Hussain FK (2014) Task-based system load balancing in cloud computing using particle swarm optimization. Int J Parallel Program 42: 739-754.