E-ISSN: 2706-8927
P-ISSN: 2706-8919
www.allstudyjournal.com
IJAAS 2022; 4(3): 224-230
Received: 18-08-2022
Accepted: 22-09-2022

Manideep Yenugula
Kroger, Blue Ash, Ohio, United States

Corresponding Author:
Manideep Yenugula
Kroger, Blue Ash, Ohio, United States

Optimizing load balancing for green cloud via efficient scheduling

Manideep Yenugula

DOI: https://doi.org/10.33545/27068919.2022.v4.i3c.1125

Abstract
The primary objective of cloud service providers is to maximize profits via the optimal use of cloud computing resources. Cloud technology is a business-oriented concept that aims at offering online IT resources as well as IT services on demand through a pay-as-you-go approach. Optimization of cloud servers on a regular basis is one of the trickiest parts of cloud computing. In order to promote the idea of green computing and increase host machine efficiency, load balancing in cloud datacenters is the major focus. In order to restore equilibrium to the data center's workload, it is necessary to use migration methods to move the virtual machines from the overloaded hosts to the less demanding host. We provide a Bacterial Foraging Optimization Algorithms (BFOA) for optimizing cloud servers, which is based on thresholds, in this study. For cloud service cost reduction, BFOA decreases the number of host systems that need to be switched on, in contrast to conventional server optimization algorithms that just take CPU, RAM, and BW use into account when scheduling resources. Through efficient use of resources, our method may help the cloud sector reduce service costs.

Keywords: Cloud computing, load balancing, dynamic resource management, server consolidation, green computing

1. Introduction
Green cloud refers to the practice of reducing energy usage by cloud data centers. Reduced trash pollution is another benefit of green cloud technology. There has been a rise in energy usage due to the rising demand for cloud computing. The migration of VMs and other server consolidation and load balancing strategies may lead to a green cloud. Virtualization provides this capability. The process involves moving the virtual machines to a different host. Performance deterioration is the overhead of migration [1]. Virtual resources are allocated dynamically in the cloud based on user demand. Workloads on the cloud's diverse resources aren't evenly distributed because of the exponential growth of data processing and storage. Servers that are overworked will take longer to finish jobs than servers that are underloaded in the same circumstance. One of the main challenges in a cloud computing system is distributing workloads evenly across the available resources [2]. In the past, cloud data centers have dissipated heat and released carbon dioxide into the atmosphere by drawing power from the grid. There is a noticeable trend towards green cloud datacenters, wherein Cloud Service Providers investigate the use of renewable energy to power their data centers in an effort to reduce carbon emissions. Solar and wind power are common renewable energy sources used by green cloud data centers [3]. This methodology executes different load scheduling as well as power management strategies. Cloud computing, which provides users with on-demand access to resources and platforms, is one of the most rapidly expanding models in the modern world. When it comes to cloud computing, load balancing is one of the toughest challenges since it has to distribute work evenly across all of the VMs on the network. Load balancing is a technique for reducing resource utilization by dividing up jobs and then dividing up the available resources across different tasks. Several writers have put forth alternative methods to improve load balancing. Recently, a technology called Osmotic Hybrid Artificial Bee along with Ant Colony has been proposed to improve load balancing in autonomous cloud settings [4]. One of the most cutting-edge and potentially game-changing technologies of the modern digital age is cloud computing. In addition, it has aided in lowering the cost of cloud provider services for small and medium organizations.

Among the most fundamental and crucial objectives of cloud computing scheduling is resource scheduling and load balancing [5]. Load balancing between virtual machines and physical machines is handled via virtualization, which a core component of cloud is computing. In PM, a heavy workload leads to an imbalanced and disorganized cloud. Virtual machine migration is a great way to manage load balance among PMs [6]. Computing in the cloud is a method wherein several users with varying needs request and receive shared resources. VMs are assigned the duties associated with application requests. Machines experience varying degrees of stress depending on the context. So, it's important to use load balancing across several virtual machines. To determine the optimal workload distribution across virtual machines, a decentralized load balancing system is used. The process of allocating workloads to VMs involves determining which VMs are most suited to execute certain tasks. The application tasks of the cloud user are then transferred to that virtual machine in order to decrease the overall execution time and energy usage [7].

A dynamic compare as well as balance method that operates on dynamic threshold levels is discussed in this study. Optimizing the data center for the cloud in order to decrease operating costs of cloud applications and optimize the cloud host computer is the purpose of this research. We provide a novel, easy-to-implement approach for optimizing cloud servers via the use of load balancing with server consolidation.

The structure of this article is as follows. We go over the basics of server optimization, load balancing, and consolidation in Section 2. Methods for distributed load balancing are briefly described in Section 3. Section 4 delves more into the construction and optimization of the cloud server. To optimize servers, we provide a paradigm that is dynamic and based on thresholds (Section 5). Our experiments along with simulation are detailed in section 6. The article is summarized in Section 7, and Section 8 delves into the topic of future work.

2. Related Work

A method for optimizing resources at the provider of cloud services is suggested in the study [8] under the name of threshold comparisons and load balancing algorithms. There are two distinct thresholds that are specified for load balancing: the lower and the higher. The basic idea behind this method is to transfer the load from one server to another based on whether its load is higher than a certain threshold or lower than a certain threshold. Virtual machines are migrated to provide a balanced load. The other hosts are turned off as the burden is concentrated on a smaller number of them. With minimal energy use, this strategy achieves the goal of making efficient use of existing resources.

In a cloud computing setting, the study examined many algorithms for load balancing, including Particle Swarm Optimizing, Artificial Bee Colony, and Grey Wolf Optimization [9]. Additionally, it aids in making better use of resources and enhancing system performance. With better convergence and simplicity, research experiment results are very promising.

To achieve load balancing and environmentally conscious energy cost reduction, the suggested method employs the nature-inspired Improved Bacterial Foraging Optimization Algorithms (BFOA), as described in [10]. To find out how well the suggested BFOA algorithm worked, simulations were run. When compared to baseline approaches, the algorithm significantly lowers the system's eco-conscious power cost.

The Grey wolf optimization method, which relies on the reliability capabilities of the resources, was used to accomplish proper load balancing in the research [11]. Here, the technique known as GWO tries to find the fitness function along with threshold for every node after first looking for busy or unemployed nodes. Results from simulations run in CloudSim show that the proposed approach not only produces optimum solutions, but also beats other methods in terms of response time and costs.

Osmotic behavior allows VMs the requirement to be transferred in the technique presented in [12]. On the other hand, the aforementioned algorithm's main flaw was that it didn't take into account resource needs while deciding how many physical machines to reduce. This is why they're migrating non-critical virtual machines, and it will have an impact on the SLA violation. As a result, a new technique called OH-BAC has been suggested to improve cloud-based load balancing and decrease virtual machine migrations. This approach is known as Future Utilization Prediction. When migrating VMs from the current pool of active physical machines, the suggested method OH-BAC-FUP takes both the current and projected resource use into account. Optimal linear regression method and linear regression are two separate forecast models that may predict the next source operation. Therefore, the regression models mentioned earlier should be evaluating how VM and PM resources may be used. After that, the OH-BAC fitness function may be employed with the predicted value to determine which VMs should be migrated to a suitable physical machine (PM).

Utilizing the idea of for cloud environments, the authors of [13] introduced a model for scheduling and balancing resources based on deep learning utilizing the MQLO algorithm. In order to identify servers that are experiencing overload and move them to virtual machines, the Multidimensional Resource Scheduling as well as Queuing Network is used. Here, artificial neural networks are used as a classifier within the context of deep learning. This allows for the identification of servers or virtual machines that are either overloaded or underloaded, and then the subsequent balancing of these resources according to their fundamental characteristics, including CPU, memory, and bandwidth. To be more precise, the success rate as well as response time have been improved using the proposed ANN-based MQLO approach. Response time, usage of energy, average success rate, as well as resource scheduling efficiency are some areas where the proposed ANN-based MQLO algorithm surpasses the present approaches, according to the simulation findings.

The article presents the authors' cloud computing system and their multi-resource load-balancing techniques [14]. The method is based on Ant Colony Optimization. A well-balanced system is maintained while the suggested method targets makespan and cost. Results from experiments conducted on benchmark processes confirm the algorithm's validity. The findings demonstrate that MrLBA decreases the execution time as well as expenses while making good use of the resources at its disposal via balancing their loads.

The article [15] presents a novel method for determining the ideal final destination PM for virtual machine migration by using a Type 2 Fuzzy Logic algorithm based on soft

computing. When moving VMs from one physical location to another, the one with the shortest migration time is chosen. The simulation is run using the CloudSim tool, with the results of the fuzzy system being handled via extensions to the Java-based fuzzy (Juzzy) packages. Performance parameters such as Transmission Time, Implementation Time, MakeSpan, as well as Number of Migrations were used to compare the simulated outcomes of the proposed VMMLB-FLS with the current Round Robin as well as CM-eFCFS cloud-based algorithms. When compared to other approaches, the findings of the suggested method show that VMMLB-FLS is more effective.

3. Proposed Work

3.1 System Model

Cloud resource managers operate on two levels, the global host level and the local host level, as shown in Figure 3. The cloud system's global load balancer is responsible for locating suitable data centers, while the local load balancer is responsible for selecting the host machine inside those data centers. Even after a virtual machine has been constructed and cloud resources allocated to a work, the job's needs could change. There has to be a way to optimize the system of clouds when there is an imbalance, which happens when the demand for any work goes up or down. The ever-changing nature of cloud applications necessitates periodic server optimization strategies. Server optimization is the focus of both load balancing as well as server consolidation. Optimizing a cloud server primarily involves making better use of available resources and enhancing the speed with which operations complete. Among the policies included in the server optimisation method are those that direct users to optimise their systems.

A. Selection Policy

In the event of an imbalance on a host node, the selection policy determines which processes must be upgraded.

B. Transfer Policy

The Transfer Strategy carries out a crucial function by deciding which virtual machines in a job may take part in the migration process and which ones cannot.

C. Location Policy

Importantly, the placement policy also finds a good host to transfer a virtual machine to so it doesn't become overwhelmed. Finding a host system that is almost as light as the nearest host nodes relative to the average weight is the primary goal of this stage. This will guarantee that after the migration, both host node are in a balanced state.

D. Information Policy

Both the means and the timing of data collection about the system's host nodes are determined by the information policy. As part of its periodic policy, every host node sends out a heartbeat signaling its existence in the system and the amount of load it is carrying at regular intervals to the rest of the cluster.

To evenly distribute demands, the suggested system employs the Improved Bacterial Foraging Optimization Algorithms (BFOA). Also taken into account were a number of basic approaches to power management and load scheduling that use both brown and renewable energy sources. Reduces data center electricity costs while being environmentally conscious.

In Figure 1 we can see the design of the system that is being suggested. From this point on, the data centre broker receives requests made by users based on their regions or bases. Virtual machine load balancers leverage the data center broker's forwarding of user requests to implement the BFOA-based algorithm. Over time, the virtual machine load balancer organizes requests across all of the cloud servers and distributes the workload evenly. Using Energy Selector, one may choose between grid power, solar power, or wind power to power the data centers.

Additionally, in order to enhance performance, the baseline technique has been modified in the following ways. Adapted method for optimizing bacterial foraging as a load balancing mechanism. Switching tasks from servers that are too busy to ones that aren't have been thought of.

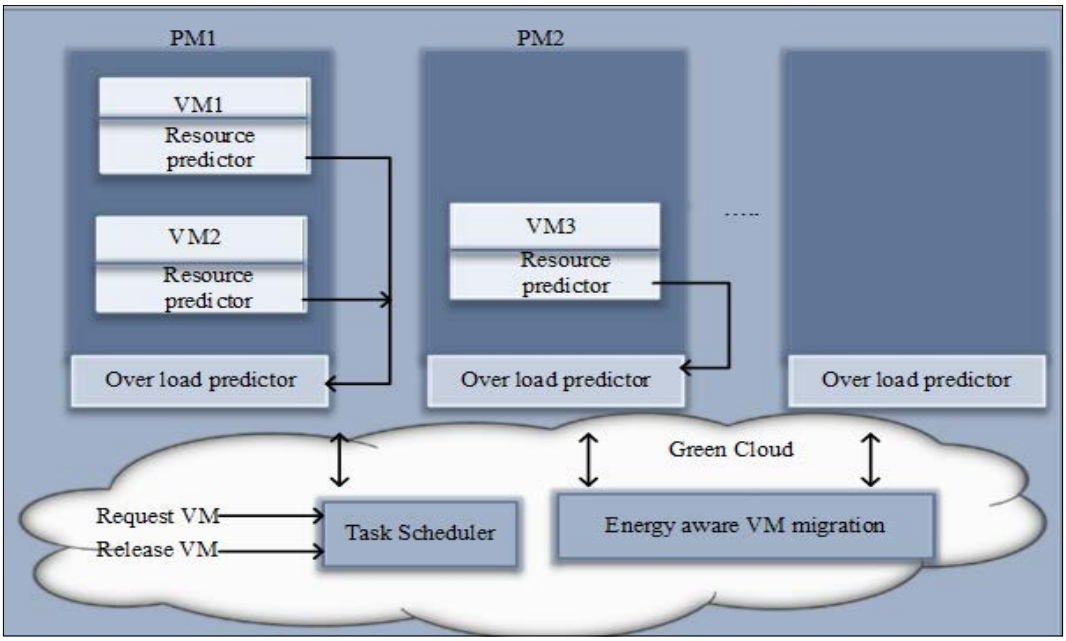

Fig 1: Load Balancing Model

3.2 Load Balancing Model

Through the use of server consolidation as well as dynamic comparison along with balancing, we laid up a strategy to enhance the cloud system. This method optimizes the servers in the cloud via process migration and tracks how each server node in the datacenter is using cloud resources to enhance load balancing along with the electrical usage of the cluster. Two principles of cloud optimization are used in this context. Two methods exist for optimizing cloud systems: one that works at the level of the host machine and another that uses dynamic threshold values determined by the actions of user applications. Assume, for the sake of argument, that a given data center has m hosts and that each host contains nj virtual machines (Vm). The total number of virtual machines in the whole information center is n.

$$n = n1 + n2 + n3 + \cdots \ldots \ldots + nj + \cdots \ldots \ldots + nm - 1 + nm. \quad (1)$$

$$H\_Load\ (Load\ of\ a\ host\ machine) = \alpha * (CPU\ Usage) + \beta * (Ram\ Usage) + \gamma * (BW\ Usage). \quad (2)$$

$$\alpha + \beta + \gamma = 1, (\alpha, \beta, \gamma)\, \epsilon\, [0\ \tilde{}\ 1] \quad (3)$$

Where α, β and γ are respectively weight coefficients.
Here, we've multiplied the total server capacity by their weight variables to dynamically construct a set of thresholds that we've utilized to assess the load.

- **Host_Limit:** Maximum allowable load on a host computer, in terms of both capacity and efficiency.
- **Upper_threshold_value_of_host (H_UTD):** The cloud provider sets the weight coefficients q based on the dynamic behavior of applications and services, and the host limit multiplied by it.
- **Lowe_threshold_value_of_host (H_LTD):** The cloud provider sets the weight coefficients * δ, where δ based on the dynamic behavior of applications and services, and Host Limit multiplied by this value is the maximum.

a) Capacity of VM: The capacity of the VM is calculated as given in equation (4).

$$C_{VM} = PE_{VM} * MIPS_{VM} + BW_{VM} \quad (4)$$

Where,
PEVM- Number of Processing Elements allocated to VM
$MIPS_{VM}$ – Millions of Instructions per Second of all PEs for VM
BW_{VM} – Bandwidth allocated to VM

b) Load on VM: The workload on VM is estimated using equation (5)

$$L_{VM} = N(t)/S(t) \quad (5)$$

Where,
$N(t)$ - Number of tasks in VM at time t
$S(t)$ - Service rate of VM at time t

c) Cost Function: The cost function J for BFOA is defined in equation (6) as,

$$J^i(j,1) = J^i(j,1) + J_{cc}\left(\Delta^i(j,1), P^i(j,1)\right) \quad (6)$$

Where,
$J^i(j,1)$ - cost of the ith bacterium in the jth chemotactic and lth elimination dispersal steps.
$J^{cc}\ (\Delta i(j,l), Pi(j,1))$ – additional cost due to the attraction and repulsion between different bacteria in the same position.
Equation (7) provides the cost functional for load balancing in this context as,

$$J^i(j,1) = L^i_{VM} + U^i_{VM} - C^i_{VM} - \left(E_i^{\ solar}(t) + E_i^{\ wind}(t) - V \right. \quad (7)$$

$$\left. \varepsilon_i^{\ grid} P_{i,\max} - E_{i,\min}^{\ solar} - E_{i,\min}^{\ wind} \right) \quad (8)$$

Where,
L_{VM} - Load on VM (under loaded) where the bacteria (cloudlet) is currently present $U^i_{VM} - CPU$ utilization at VM

C^i_{VM} - Capacity of VM

What follows is an explanation of the fundamentals and procedures of an optimization strategy for bacterial foraging, a modified algorithm for bacterial foraging, and the variables taken into account by different power management and load scheduling strategies.
The BFOA is an approach to swarm intelligence that uses a distributed optimization method to mimic the collective and individual foraging strategies of E. coli bacteria. According to this hypothesis, animals seek for food and nutrients in a way that optimizes the amount of energy they consume per unit of time, which is the foundation of the foraging notion. Bacteria are able to interact with one another because they use signals. Here are the stages involved in the optimization process.

- Chemotaxis - small steps (swim, tumble) taken by the bacterium in search of nutrients
- Swarming - arrange themselves into a concentric pattern (group)
- Reproduction - healthy bacterium split into two
- Elimination-Dispersal - few bacteria are eliminated and randomly dispersed in the optimization domain

In this case, the bacterium's location yields the optimization process's answer. Starting as a group, the solution eventually converges on the best possible answer. What follows is a synopsis of the updated algorithm.
Modified BFOA is started at every new slot to move cloudlets from overcrowded to underloaded virtual machines (VMs), which helps with load balancing in the system. Here is the cloud infrastructure and bacterial map. Overloaded virtual machines (VMs) release cloudlets, which are like germs that hunt for underloaded VMs to eat. Given that BFOA is concerned with bacterial foraging and nutrient search in a virtual search distance, the method may be used to determine which underloaded virtual machine is the most suitable for migrating the cloudlet. The reproduction loop is removed from the original BFOA because it only allows a certain amount of clouds from overloaded virtual machines to reproduce, which is the healthier half of the bacteria. This means that the other halves of the cloudlets are rejected. We

may decrease the overhead caused by the deletion of the reproduction loop by considering the location of a cloudlet in relation to all other cloudlets.

Below, we outline the procedures that will be engaged in the planned BFOA. Underloaded virtual machines are located and assessed in the chemotaxis stage. The swarming stage involves grouping virtual machines (VMs) according to their cost in the chemotaxis process. At last, the elimination procedure verifies whether the present condition of virtual machines has changed and determines if the cloudlet should be moved to another virtual machine. Let $E^{grid}(t)$, $E^{solar}(t)$, $E^{wind}(t)$ denote the total energy that data centers get at time slot 't' from the grid, solar power, and wind power, respectively. ε^{grid} and ε^{ren}, where P(t) is the cost of grid energy and W(t) is the workload at timeslot 't, and are eco factor numbers for renewable and grid energy sources. Here, "N_c," "N_s," and "N_ed" stand for the number of steps taken during chemotactic, swimming, and elimination-dispersal mechanisms, respectively. As indicated, the load-related threshold is T_s. Algorithm 1 displays the BFOA specifics.

Algorithm:1 BFOA load balancing

Input: $E^{grid}(t), E^{solar}(t), E^{wind}(t), \varepsilon^{grid}, \varepsilon^{ren}, P(t), W(t)$ (9)

Output: Load balancing strategies

1. Start
2. While simulation not terminated, do at each time slot,

a) Compute standard deviation of loads across all VMs

b) If $\sigma < T_s$ system is balanced, stop

c) If total_load > total_capacity

d) Enqueue tasks d. Initialize colony of bacteria with cloudlets from overloaded VMs

e) Compute initial costs

f) While $N_{ed} > 0$, do

g) While $N_c > 0$, do

- Tumble to new VM
- If variance (new costs) < variance (old costs)
- While $N_s > 0$ and variance reduces, do

i) Try other VMs in same datacenter

ii) Update colony

Continue

1. Assign random P_{ed} to all bacteria
2. Randomly disperse bacteria based on P_{ed}
3. Migrate cloudlets to new positions (VMs) based on bacterial positions

Stop

Chemotaxis is a process that mimics the movement of an E. coli cell. Bacteria may swim in a straight line for long periods of time when conditions are good, or they can move in a zigzag pattern when conditions are bad. In computing chemotaxis, equation (1) is used. Regarding this matter,θ^i (j,1) denotes the location of the i^\"th" bacterium at the j^\"th" chemotactic stage and the 1^\"th" elimination-dispersal phase.

$$\theta^i(j+1,1) = \theta^i(j,l) + C(i) * \left(\Delta(i)/\left(\sqrt{}(\Delta^T(i) * \Delta(i))\right)\right) \quad (10)$$

Where,

$\theta^i(j+1,k,l)$ – position of bacterium i in $j+1^{th}$ chemotactic step and 1^{th} elimination step.

$\theta^i(j,k,1)$ – position of bacterium i in j^{th} chemotactic step and 1^{th} elimination step.

C(i) -step size in random direction.

Δ (i) - random unit direction vector $(-1,1)$.

Bacteria or cloudlets undergo cost evaluation relative to their location in the population at each chemotactic step, which involves falling to a new underloaded virtual machine. To swim, or attempt other underloaded virtual machines (VMs) in identical data center, a bacteria must first determine if its new location's cost is cheaper than its old one. Factoring in the bacterium's cost at that location as well as its attractive and repulsive properties throughout the swarming phase yields its total cost.

To mimic the unintentional shifts that occur in the bacterium's feeding habitat, the elimination dispersion stage is used. This has the potential to relocate certain germs to a different area. While the elimination dispersion phase may halt chemotactic advancement (equation (2)), it can also aid in reaching the global optimum independently of local optima (equation 11). Here, we examine the probability P_ed given to each bacterium/cloudlet to see whether it should be distributed to a new site.

$$\theta^i(0,1+1) = \begin{cases} \theta^i(j,l) + \left(\Delta(i)/\sqrt{}(\Delta^T(i)^*\Delta(i))\right) \\ \text{if } P(j,l) >= P_{ed} \\ \theta^i(j,l) \text{ if } P(j,l) < P_{ed} \end{cases} \quad (11)$$

Additionally, while developing plans for power management and load scheduling, factors such as virtual machine capacity, virtual machine load, and the cost of function are taken into account.

Load Balancing Cases

- **Case 1:** Determining a circumstance where load of host is larger than the preset upper threshold value. The host is deemed to be overloaded in this instance. In order to distribute the additional workload from this host to a different computer, a load balancing method must be used. An adaptive comparison and equal load balancing strategy accomplishes this task. Another host computer is located here, and it has a p[k] chance of having the fewest virtual machines that can handle this workload without going down.
- **Case 2:** If the host load is more than a certain minimum amount, an event will be identified. The host is deemed to be underloaded in this instance. In order to conserve energy and reduce the cost of cloud services, we need to use a server consolidation method to move the host's workload to another machine. Then, we may turn off the host. The migration of virtual machines is used to accomplish this task. We have located an alternative host computer that can handle this workload without experiencing overload.

4. Results and Discussion

To practice, we utilize the cloudsim toolkit, which is a program for creating virtual machines on the cloud. Host, data center, application, and other layers may all benefit from load balancing with server consolidation algorithms.

Using the cloudsim simulator, we implemented it at the host level in a cloud environment. When a user requests a virtual machine to do a job, the host computer enables the request and makes available the necessary resources. At the virtual machine level, the cloudsim simulator makes use of VmScheduler, which operates on two policies: TimeShared and SpaceShared. Time and physical space are two resources that these strategies aim to facilitate. At the virtual machine level, cloudletScheduler uses the same two policies-TimeShared and SpaceShared-to distribute resources among the cloudlets operating on each VM. A broker called DatacenterBroker connects virtual machines to host computers in cloudsim. We model our method with a range of lower and higher limit values for the thresholds since our algorithm relies on these values, which are determined by the application's behavior.

Apache Commons Mathematical Library and NetBeans IDE were both incorporated into the CloudSim simulation platform. CloudSim includes necessary classes for modeling cloud infrastructure, including data centers, hosts, virtual machines, applications, consumers, and rules for managing tasks like provisioning and scheduling. We validated the method in the cloud data warehouse environment using the simulation setup settings listed in Table 1.

Table 1: Simulation parameters

Parameters	Values
No of datacenters	20
No of hosts per datacenter	5-10
No of VMs	100
No of cloudlets	500
Number of PEs / VM	4
Number of MIPS / VM	2500
Length of time slot	200 ms
Solar conversion efficiency	20%
Wind conversion efficiency	30%
Total solar irradiation area	15000 m2
Total rotor area	25000 m3
Density of Air	1.225 kg/M3
Eco factor of renewable energy (ε^{ren})	1.0
Eco factor of brown energy (ε^{grid})	9.0
T	0.85
Nc	20
Ns	5
Ned	0.25
Ped	

The BFOA algorithm's performance analysis is shown in the following discussion section.

We ran simulations to see how the BFOA fared against some of the current gold standards, such as the cheapest first and quality of service first approaches. A dataset including real-time parallel workloads was used to conduct the simulations. We measure the BFOA's efficiency by looking at its latency and energy usage.

Table 2: Performance using BFOA

EC (mJ)	ET (ms)	LT (ms)	TH(kbps)
0.9	0.09	0.001	50
6.5	0.28	0.029	467
13	0.67	0.09	679

From the results of the outstanding performance green clouds system, it is clear that the suggested method is working. A small limitation in data centers is the efficient use of electricity by the workstations or virtual machines. To run the resources that need load balancing, power consumption will rise in direct proportion to the exponential growth in resource demands. Low energy usage, the intended output, and the effective execution of all jobs on the machines with load balancing are both achieved by the suggested technique. In a green cloud environment, important resources can't be wasted if computers are overloaded and explode. Power consumption shouldn't be too high to avoid this. When it comes to cloud computing, latency is a key performance indicator. There ought to be little delay. As it rises, more requests or tasks will have to wait in line, which may lead to queue overflow and the reduction of the number of requests needed for resource allocations. This leads to an undesirable outcome—an increase in the strain on the green cloud operations and a subsequent degradation in performance. Figure 6 displays the proposal's latency, which demonstrates that the hybrid method is successfully executing operations in the green cloud environment with minimal latency. How many resource allocation procedures pertaining to the overall amount of tasks allocated to the virtual machines for execution are successfully handled by the planned work's throughput? The throughput is crucial because it determines how much energy the machines consume. If the throughput drops, the machines will be overwhelmed and request execution will slow down. Hence, a high throughput is required.

5. Conclusion

One of the hardest aspects of cloud computing is routinely optimizing cloud servers. The main goal of load balancing in cloud datacenters is to boost host machine efficiency and advance the concept of green computing. Virtual machines must be moved from the overworked hosts to the less demanding host using migration techniques in order to bring the burden in the data center back into balance. In this work, we provide a threshold-based Bacterial Foraging Optimization Algorithms (BFOA) for cloud server optimization. BFOA reduces the number of host systems that must be turned on in order to reduce cloud service costs. This is in contrast to traditional server optimization methods, which only consider CPU, RAM, and BW utilization.

6. References

1. Jeyakarthic M, Subalakshmi N. Client Side-Server Side Load Balancing with Grasshopper optimization Mapreduce Enhancing Accuracy in Cloud Environment. 2020 Fourth International Conference on Inventive Systems and Control (ICISC); c2020. p. 391-395.
2. Kaur A, Kaur B, Singh P, Devgan MS, Toor HK. Load Balancing Optimization Based on Deep Learning Approach in Cloud Environment. International Journal of Information Technology and Computer Science. 2020;12:8-18.
3. Adaniya A. A Proposed Load Balancing Algorithm for Maximizing Response Time for Cloud Computing Environment using Fusion Algorithm. International Journal for Research in Applied Science and Engineering Technology; c2019.

4. Talaat FM, Saraya MS, Saleh AI, Ali HA, Ali SH. A load balancing and optimization strategy (LBOS) using reinforcement learning in fog computing environment. Journal of Ambient Intelligence and Humanized Computing; c2020. p. 1-16.
5. Radhamani V, Dalin G. Selection of migration vms and destination pms using an optimization algorithm in Pca-Ta-Irial approach for green and load balanced cloud computing; c2020.
6. Kiruthiga G, Vennila SM. Energy Efficient Load Balancing Aware Task Scheduling in Cloud Computing using MultiObjective Chaotic Darwinian Chicken Swarm Optimization; c2020.
7. Liu X. Optimization of Load Balancing Scheduling Model for Cloud Computing Resources in Abnormal Network Environment. Journal of Advanced Computational Intelligence and Intelligent Informatics. 2019;23:356-361.
8. Shahapure NH, Rekha PM, Poornima N. Threshold Compare and Load Balancing Algorithm to for Resource Optimization in a Green Cloud. Revista Gestão Inovação e Tecnologias; c20121.
9. Gohil BN, Patel DR. A hybrid GWO-PSO Algorithm for Load Balancing in Cloud Computing Environment. 2018 Second International Conference on Green Computing and Internet of Things (ICGCIoT); c2018. p. 185-191.
10. Thilagavathi N, Subha R, Rhymend Uthariaraj V. Eco-Aware Load Balancing for Distributed Cloud Data Centers with Renewables. 2018 Tenth International Conference on Advanced Computing (ICoAC); c2018. p. 229-236.
11. Sefati S, Mousavinasab M, Zareh Farkhady R. Load balancing in cloud computing environment using the Grey wolf optimization algorithm based on the reliability: performance evaluation. The Journal of Supercomputing. 2021;78:18-42.
12. Tuli K, Kaur A. Load Balancing Scheme for optimization of Virtual Machine Migration using swarm in Cloud Environment. 2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO); c2021. p. 1-6.
13. Deep learning based load balancing using multidimensional queuing load optimization algorithm for cloud environment; c2020 Oct.
14. Muteeh A, Sardaraz M, Tahir M. MrLBA: multi-resource load balancing algorithm for cloud computing using ant colony optimization. Cluster Computing. 2021;24:3135-3145.
15. Negi S, Rauthan MS, Vaisla KS, Panwar N. Efficient Load Optimization Method Using VM Migration in Cloud Environment; c2021.