International Journal of Advanced Science and Research
www.allsciencejournal.com
ISSN: 2455-4227
Received: 28-04-2021; Accepted: 14-05-2021; Published: 29-05-2021
Volume 6, Issue 2, 2021, Page No. 34-41

Load balancing in cloud computing using Multi-objective Bio-inspired techniques - A comprehensive study

Brototi Mondal
Assistant Professor, Department of Computer Science, Sammilani Mahavidyalaya, University of Calcutta, Kolkata, West Bengal, India

Abstract
A significant issue in cloud computing is load balancing, which makes it difficult to guarantee the appropriate operation of services related to Quality of Service, performance evaluation. Mapping workloads to utilize computer resources that considerably increase performance is the main goal of load balancing. Due to the huge solution space, load balancing in cloud computing is classified as a "NP-hard" topic. As a result, predicting the ideal outcome takes more time. There aren't many strategies that might possibly produce a polynomial-time optimal solution to these problems. For those sorts of problems, it has been demonstrated in prior studies that metaheuristic-based techniques may produce correct results in a respectable amount of time. Based on performance metrics such as makespan time, degree of imbalance, response time, data center processing time, and resource consumption, this article offers a comparative analysis of several metaheuristic load balancing methods for cloud computing. Based on performance criteria, it demonstrates how different Meta-heuristic Load Balancing approaches perform. Numerous approaches including evolutionary and non-evolutionary algorithms have been suggested to develop efficient procedures and algorithms for allocating the clients's requests to reachable Cloud nodes. This study's objective is to investigate various load balancing algorithms which reduce response time while maximizing resource use. The assessed algorithms are categorized in accordance with a thorough taxonomy that is focused on load balancing techniques in cloud data centers. The different techniques are demonstrated and compared also.

Keywords: Cloud computing, load balancing, Bio-inspired, meta heuristic

Introduction

Using a pay-as-you-go subscription model, cloud computing offers customers access to infrastructure, platform, and software as services. It offers consumers a variety of networking, processing, and storage resources over the internet, enabling users to store a lot of data and access a lot of computing power using their PCs. Cloud computing provides considerable benefits to IT firms by relieving them of the low level labor of building up fundamental hardware and software infrastructures so that they can concentrate more on innovation and the production of business values [1]. Cloud service provider (CSP) offers computing, software and storage as service. Because on-demand provisioning and de-provisioning enables organizations to lower capital expenditures for software and hardware, it has gained widespread adoption. The CSPs must rent out processing capacity to customers in the form of virtual machines (VMs) as the size of the cloud may fluctuate. Up scaling or maintaining the same performance is the fundamental hurdle for cloud service providers. Numerous new cloud-based applications, such as those for social networking, web hosting, content delivery, and real-time instrumented data processing, have diverse needs for deployment, composition, and setup. The three main issues with the Cloud computing are the virtualization problem, Distributed architecture and the load balancing issue. Load balancing is one of these challenges where we must evenly spread the local workload throughout the entire cloud and make sure that at any given moment, every processor or resource in the cloud is performing roughly the same amount of work [3]. In order to correctly allocate a load depending on the computing power and bandwidth of each service, it must first travel by scheduling machines as it reaches the cloud.

It is extremely difficult to quantify the effectiveness of scheduling and allocation strategies in a real Cloud environment for various application and service models under numerous circumstances because (i) clouds display variable demand, supply patterns, and system sizes, and (ii) users have heterogeneous and conflicting QoS requirements [2].These circumstances can easily result in imbalanced loads in cloud data centers, which can also cause performance deterioration and SLA breaches. To address these issues, a load balancing mechanism is necessary on cloud data centers to boost the performance of the cloud system of the cloud service provider, whether through maximizing computing resources or in any other way. By doing this, the situation where some resources are overburdened while others are dormant or performing minimal work is avoided. An effective load balancing mechanism should be dynamic and adaptable to the changing environment to satisfy the requirements [4]. A review of the most recent load balancing methods, designed especially for use in cloud computing systems is demonstrated in this work. Additionally, the response times of different algorithms are contrasted.

1. Cloud system setup

These algorithms use datacenter (DC), virtual machine (VM), host, and cloudlet components to implement a suggested algorithm. Requests for services are handled by a datacenter component. Since VMs are made up of application components that are related to these requests, data centers should host the VMs that users request. VM

creation, VM destruction, and VM migration are the first steps in the life cycle of a virtual machine. The following provides a succinct explanation of these elements and how they interact:

a. **Datacenter:** Numerous hosts with homogeneous or heterogeneous configurations of memory, cores, capacity, and storage are present in each datacenter. Additionally, it generates the allocation of bandwidth, memory, and storage devices.

b. **Virtual Machine:** Memory, processor, storage, and VM scheduling policy are all VM characteristics. On a single host, several virtual machines can run concurrently while maintaining the processor sharing policy. The research work takes into account the requirements for VMs with uniform configuration.

c. **Physical Host:** The datacenter contains hosts that are either homogeneous or heterogeneous in their configuration. In this experiment, VMs are expected to be able to process a large number of cores, and the host should have a resource allocation policy to distribute these resources among the VMs. Therefore, the host can set aside enough memory and bandwidth for the process elements to run inside the VM. VM creation and destruction are handled by the host.

d. **Cloudlet:** A cloudlet is an application task that runs on a virtual machine that is mounted on a host and is in charge of delivering data in the cloud service model.

e. **VM Provisioner:** It is employed to assign a host to a required VM. In cloud analysis, virtual machines (VMs) are mapped to virtual machines at random, and load balancers choose which VMs to process requests on based on load balancing policy.

2. Need for Load balancing in cloud

Workload balancing is a crucial component of cloud computing architecture that aids in the distribution of computer resources. Processing power, memory, and storage capabilities vary from VM to VM. The only method to perfectly match a task with a VM so that no VM may be overwhelmed is through load balancing. The web's dynamic computing causes a cloud architecture to experience request overload. The most challenging and crucial field of study in cloud computing is load balancing, which is used to distribute workloads among virtual machines (VMs) in data centers. The primary idea behind cloud computing is the sharing of resources as needed across the internet. Interconnected computer devices, storage, and data centers are essential elements of cloud computing [5]. To share data, software, hardware, and computing resources with other devices, cloud computing employs a distributed and parallel computing method. The "pay-per-use" model is offered by this model. To complete a job, a client doesn't need to buy any computational platforms or software; instead, all they need is an internet connection to access the cloud services and computing resources, and they just pay for the services they utilize. It lowers the cost of purchasing a software package that is not used constantly and permits the dynamic usage of resources that many users can access at once without lowering the quality of the service. Due to the following causes, cloud service providers encounter difficulty in providing a high-quality service:

- The size and complexity of the public cloud
- The potential weaknesses of conventional load-balancing algorithms
- The variation of key stakeholders whose function is to perform customer queries

In heterogeneous cloud computing, load balancing is necessary to ensure the highest level of service and best resource usage. In order to maximize resource usage, satisfy customers, and do so at the lowest possible cost, load balancers aid in the equitable distribution of resources to workloads. The load-balancing techniques now in use have a number of problems that require rapid correction. It encourages researchers to develop better load-balancing strategies to get around these challenges [7]. The fundamental way that metaheuristic-oriented strategies get over these problems is by providing precise answers in an acceptable amount of time. Due to its performance, dependability, quality of service, and effectiveness in solving significant and complex obstacles, metaheuristic load balancing has garnered more attention in recent years. For those sorts of problems, it has been demonstrated in prior studies that metaheuristic-based techniques may produce correct results in a respectable amount of time.

3. Load balancing algorithms in cloud computing

Cloud load balancing is a method that evenly divides the extra dynamic local workload among all of the nodes. In cloud computing, VM scheduling with load balancing seeks to allocate VMs to appropriate hosts and optimize resource use across all of the hosts [5]. The optimal use of the resources that are already available can be achieved by implementing effective load balancing algorithms, which will reduce resource usage. Optimizing the use of cloud computing resources, such as CPUs, storage, and memory, requires load balancing. The task of allocating and utilizing resources is carried out by virtual machines operating on real hardware. When workloads are executed on VMs, some of the VMs may be overused or underused. Each computer in the cloud data center will carry out the same amount of jobs at any one moment according to their capability thanks to load balancing mechanisms. Because user needs in cloud computing are so dynamic, multi-tenancy necessitates segregating various users inside the cloud architecture. To distribute load across virtual machines, accomplish optimal use of cloud resources, and improve performance, several heuristic and metaheuristic approaches were applied by various researchers in existing cloud computing research.

Workload mapping on vast computer resources in cloud computing is a classification problem characterized as a "NP-hard" problem. For such problems, no optimization technique can produce the best solution in polynomial time. Technically speaking, solutions based on in-depth analysis are not practical since creating work plans has such a large operational cost. The main goal of the load-balancing technique is to evenly distribute workloads among virtual machines and processing power in order to reduce relative imbalance [21]. Heuristic and Meta-heuristic approaches are frequently employed in cloud computing to provide load balancing. These techniques offer a number of crucial characteristics, such as a larger search space with random

searching that aids in finding the best solution to a scheduling problem in a predetermined amount of time.

The metaheuristic algorithm has a greater computing cost than the heuristic approach. To increase the convergence rate of metaheuristic approaches, the majority of researchers use heuristics techniques that condense the search space. In this procedure, there are numerous goals [22]. Figure 2.

depicts different meta- heuristic algorithms which are widely used for load balancing in cloud.

For a conventional load balancing architecture used in a cloud environment, the load balancer balances the load by distributing the demand proportionately among the underloaded VMs indicated in Figure. 1.

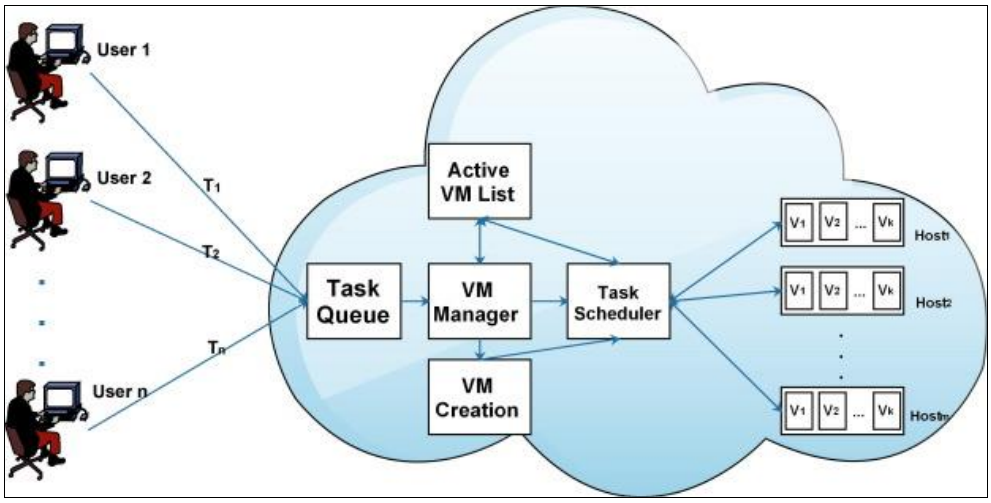

Fig 1: Load Balancing in Cloud

The two basic types of load balancing are static and dynamic load balancing,

1. **Static Load balancing:** Systems with extremely low load variation are designed with static algorithms. The static method equitably distributes all traffic among the servers. For improved processor performance, this algorithm needs in-depth knowledge of server resources, which is decided at the start of development. Load shifting decisions are made independently of the system's existing status. Static load balancing algorithms have a number of significant limitations, one of which is that load balancing jobs only function once they have been generated.

Dynamic approach: In order to balance the load, the dynamic algorithm first identifies the lightest server in the entire network and gives it priority. This necessitates real-time network connection, which might raise system traffic. Here, the load is managed based on the system's present status. Making load transfer decisions based on the current system state is a property of dynamic algorithms. In this system, processes can be transferred in real time from a machine that is frequently used to a machine that is rarely used.

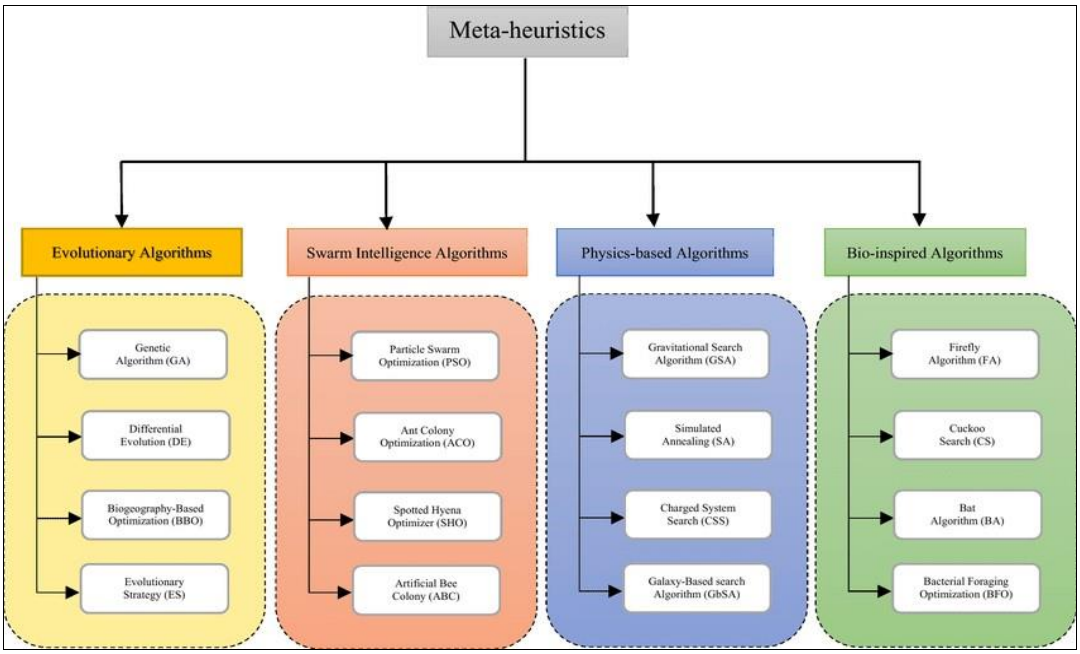

Fig 2: Meta heuristic Algorithms for Load Balancing

1. Load Balancing Process

The following actions are a part of the load balancing process:

a. **Determining the user task needs:** This stage defines the resource needs of the user tasks that will be allocated for execution on a virtual machine

b. **VM resource facts detection:** This verifies a VM's resource information state. It provides the unallocated resources and the virtual machine's current resource use. Based on this stage, the condition of VM with regard to a threshold can be identified as balanced, overloaded, or under-loaded.

c. **Task scheduling:** A scheduling algorithm schedules the tasks to the right resources on the appropriate VMs once the resource specifics of a VM have been discovered.

d. **Allocation of resources:** A resource allocation strategy is being used to allocate the resources to run the scheduled tasks. There are numerous scheduling and allocation policies put forth in the literature. The performance of the scheduling algorithm and the allocation strategy influence the load balancing algorithm's robustness.

e. **VM migration and task migration:** In the process of load balancing in the cloud, migration is a crucial step. To solve the overloading issue, VM migration involves moving a virtual machine (VM) from one physical host to another. It can be divided into live and non-live migration types. The movement of tasks between VMs is known as task migration, which has two types: intra-VM task migration and inter-VM task migration. There have been many different migration strategies put forth in the literature. An effective load balancing method follows from a successful migration strategy.

2. Challenges of load balancing algorithms in cloud

Algorithms for load balancing are used at both the applicatio n and VM levels. The load balancing algorithm can be integrated into VM manager at the virtual machine level and into application scheduler at the application level. The following are the difficulties and main challenges when making an effort to recommend an optimal solution to the problem of load balancing in Cloud Computing [6]:

a. **Cloud Nodes' Spatial Distribution:** Some algorithms are only intended to be effective for close-proximity nodes where communication delays are minimal.But developing a load balancing algorithm that can function for spatially distributed nodes is difficult. This is due to the need to consider additional factors, such as the speed of network links between nodes, the distance between the client and nodes that process tasks, and the distances between nodes that are involved in service provision [7]. To effectively tolerate long delays, a load balancing mechanism that can be controlled across all of the spatially distributed nodes is required.

b. **Performance:** Users' satisfaction and experience can indicate performance. The performance can be evaluated by considering some parameters like, resource utilization, scalability, response time, processing time, throughput etc. The off-loading of overloaded hosts with higher resource utilization is recommended by various VM load balancing algorithms. Whereas scalability indicates that even with an increase in users, the service quality remains consistent. Response time can be described as how long a load balancing algorithm in a cloud system takes to respond. This parameter needs to be decreased for improved performance. Processing time is length of time needed for a task to be completed which should be minimum. Throughput gauges how quickly hosts can respond to requests because unbalanced loads can impair performance of the system.

c. **Overhead:** It establishes the level of overhead associated with the implementation of a load balancing system. It consists of overhead from communication or VM migration costs. Overhead should be reduced by a well-designed load balancing algorithm.

d. **Complexity of Algorithm:** The algorithms which need more data and communication to monitor and control the system, delays will become more problematic and efficiency will decrease. A complicated technique would result from the higher implementation complexity, which could have an adverse effect on performance. So, less complicated load balancing algorithms are preferred for both implementation and application.

e. **The failure point:** Its goal is to make the system better so that the provisioning of services is unaffected by a single point of failure. Load balancing algorithms should be developed to address this issue because, in a centralized system, if one central node fails, the entire system would also fail.

3. Problem Statement and definitions

As cloud computing is gaining popularity day by day, it is possible for some of the VMs to be overused and others are underused when tasks are scheduled on virtual machines (VMs) [8]. The distribution of a system's total load among the involved VMs using load balancing has long been recognized as an efficient technique.

Definition 1: Virtual machine (*VM*).

A virtual machine can be defined as a vector *VM,*

$$VM = \{idvm, MIPS, ram, bwd, pnum\} \quad (1)$$

where*idvm* is the unique identification number of a VM, *MIPS*(million instructions per second) is the processing speed per processing element (*PE*) of a VM, *ram* indicates RAM of a VM, *bwd* describes the bandwidth of a VM, *pnum* is representing the number of PE in a VM.

Definition 2: Physical Host (*PH*).

A physical host can be represented as a vector *PH,*

$$PH = \{hid, MIPS, rm, bwh, pnumber\} \quad (2)$$

where *hid* is the identifier number of a *PH*, *MIPS* (million instructions per second) indicates the processing speed per processing element (*PE*) of a PH, *rm* defines RAM of a PH, *bwh* represents the bandwidth of a PH, *pnumber* is the number of PE in a PH.

In clouds, load balancing may occur between physical hosts and virtual machines. The dynamic workload is evenly distributed among all the nodes by this balancing mechanism (hosts or VMs).

4. Load Balancing Metrics

The load balancer must evenly distribute the workload across all of the available resources in order to achieve better resource utilization and improved system performance. To improve the performance of the entire system, all the following factors must be managed effectively at the time of choosing load balancing algorithm:

a. ***Makespan:*** This is the total amount of time taken to allocate resources to its customers.

b. ***Throughput:*** This is the rate of processing submitted user request in a given amount of time. Greater system performance is achieved with higher throughput rates.

c. ***Scalability:*** This quantifies the node's capacity to maintain consistent load balancing even when the required number of nodes increases.

d. ***Fault Tolerance*****:** This gauges how well the load balancing technique continues to work even in the event of a node or link failure. This parameter indicates whether or not an algorithm can tolerate difficult errors. It makes it possible for an algorithm to keep working appropriately in the case of a failure. If the algorithm's performance declines, the decline is proportional to how bad the failure was. Load balancing can completely fail even with a little error.

e. ***Response Time:*** This is the total duration of time taken to respond to a submitted request which need to be minimized.

f. ***Migration Time:*** This is the total amount of time required to transfer a request from an overloaded machine to an underloaded machine. System performance increases as migration time's decrease.

g. ***Resource Utilization:*** Resource utilization is used to determine whether a host is overloaded or underutilized. Utilizing resources involves automated load balancing. Unexpectedly many processes in a distributed system may call for extra processing capacity. It is more effective to transfer resources to processors that aren't fully utilized by the program.

h. ***Stability:*** A load balancing algorithm's advantages in terms of achieving quicker performance in a certain length of time and the delays in the transfer of information between processors may both be used to describe stability.

i. ***Process Migration:*** The process migration parameter demonstrates a node's ability to move a running process to another node. Whether to construct it locally or on a distant processing element is decided. The algorithm is capable of determining whether to adjust the load distribution while the procedure is running or not.

Bio-inspired Algorithms in Cloud Computing

Metaheuristic approaches are employed instead of standard load balancing techniques because they are more effective for less severe uncertainty concerns. It's a heuristic method that is independent of the difficulty of the issue. A meta-heuristic technique is an interactive formation mechanism that directs the use of the search space and the exploration process. One technique that may be used to address performance difficulties, such as work scheduling, is the use of meta-heuristic approaches [24]. There are two primary groups of metaheuristic methods: (1) those based on local searches and (2) those based on random searches.

Need for meta-heuristic algorithms

An objective function can be optimized with the use of a metaheuristic method. It may be used internally to address a variety of optimization problems, load balancing being one of them. The necessity of metaheuristic techniques is covered in this paragraph.

Heuristic: By sacrificing the ideal answer, precision, accuracy, or speed, heuristic approaches solve problems more quickly and easily than conventional methods. Heuristic techniques are mostly used to solve NP-complete problems, a class of complicated problems [24].

- Its design is typically problem-focused.
- It is straightforward to get stuck at local optima.

Metaheuristic

These procedures resemble the heuristic technique. These methods rely on two different characteristics. The number of feasible solutions used in each recursive iteration is the first characteristic. The usual preliminary solution is used as the starting point, and alternative answers are substituted for it at each stage of the quest. It is necessary to use a metaheuristic for the reasons listed below:

- A meta-heuristic method is suitable for a wide range of challenges.
- Suitable for Multimodal Optimization problems.
- Acceptable to discontinuous, nonlinear functions.

Review of Load balancing algorithms

Numerous studies of load balancing techniques in the cloud environment have been conducted over recent years, and they have provided a strong framework for comprehending the various facets of this problem. This section summarizes earlier survey studies.

Regardless of the current load on the node, Minimum Execution Time (MET), Minimum Completion Time (MCT), Min-Min and Min–Max is employed to allocate each job in any order to the nodes where it is anticipated to be executed the quickest by minimizing the makespan and energy Consumption. [9]. A round-robin algorithm distributes tasks evenly among all processing units or data centers.

In a dynamic cloud environment, Ren proposed an algorithm known as ESWLC based on WLC (weighted least connection) that considers the node capacity when allocating resources to the least weighted task [10]. Based on each node's weight and capacity, the task is deployed.

Dasgupta *et al.* [11] proposed the genetic algorithm for load balancing in a cloud computing environment. The three phases of the genetic algorithm are selection, crossover, and

mutation. The selection of a virtual machine (VM) based on the cost and execution time of the scheduled tasks is the first step of the algorithm. To perform the crossover between the VM and the scheduled tasks, the second phase entails calculating the best costs and times. Then the algorithm mutates the scheduled tasks and the available VM before it can finally be accepted for execution.

S. Dam *et al*. [12] developed a load balancing strategy in cloud computing based on metaheuristic algorithm ant colony optimization which is basic foraging behavior of an ant that encouraged them to find the optimal shortest path from their nest to food. Here a new request is scheduled to VMs according to FCFS scheduling policy but if VMs are not available to allocate next job a random number of ant with same pheromone value is created and placed randomly to traverse. An ant chooses a VM then check whether the ant completes its tour or not. If the tour is completed then the pheromone value is updated. This is continued until optimal VM is found.

Mondal *et al*. [13] proposed a local optimization algorithm stochastic hill climbing based load balancing technique for solving work load balance issue in cloud environment. In this paper, the authors suggested to maintain an index table for VMs containing the states of the VMs which is either BUSY or AVAILABLE. Initially it is assumed that all VM's are available. When a request is arrived, a query is generated for the allocation to VM. If the VM is found available then the request is allocated to the VM and allocation table is updated accordingly. But if no appropriate VM is available to assign, a random VM id is generated and the task is assigned to that randomly generated VM. When the VM finishes processing the request and the response cloudlet is received. Generate a notification of VM de-allocation.

A load balancing algorithm for cloud computing that was inspired by honey bee behaviour was proposed by Babu *et al*. in 2013 [14]. According to this method, load balancing in cloud computing is accomplished by simulating honey bees' search behaviour. This algorithm examines and takes advantage of honey bee behaviour to locate and gather food resources. A class of bees in a honey bee colony known as scouts looks for food sources. One of the scout bees returns to the hive after finding food to inform the colony as a whole by engaging in a "vibration dance," which gives the other bees a sense of the type and quantity of the food as well as how far it is from the hive. Bees that are foraging follow the scout to the source of the food and gather it. Virtual machines represent the food resources, and tasks represent the honey bees. A task being loaded on a virtual machine is comparable to honey bees searching for a food source. A virtual machine that is overloaded is like trying to get honey out of a food resource that isn't there. In the same way that looking for new food resources necessitates removing the overloaded machine, this calls for moving the task to an underloaded virtual machine. Transferring the removed tasks to a suitable underloaded machine from the overloaded virtual machine is advised.

Using a cluster-based load balancing technique, Kang *et al*. [15] presented multimedia loading in 2016. The suggested method consists of two levels. They first build clusters at the first level to monitor requests and resources, and they base their QoS level there on greeting their neighbours. At the next level, they decide whether to transfer requests based on the average waiting time. Even though this method's performance can be improved in a real-world setting, packet loss can still cause congestion.

In order to decrease makespan time, Ebadifard *et al*. [16] provided a method for scheduling requests in 2017. They used the load balancing algorithm in the cloud platform. In order to improve load balancing and decrease response times while increasing virtual machine utilization, Ebadifard *et al*. [17] proposed a new scheduling method in 2017.

Atul Vikas Lakra *et al*. [18] described how underutilization of cloud resources results from poor task scheduling in the cloud and developed a multi-objective task scheduling algorithm in this regard to optimize throughput and reduce execution costs while maintaining SLAs. The suggested scheduling method assigns tasks a priority based on their QoS value, giving lower priority to tasks and vice versa. On receiving a list of VMs from cloud service providers, cloud broker assigns QoS to the VMs.

Jing Liu *et al*. [19] proposed the multi-objective genetic algorithm (MO-GA), which emphasized encoding rules, crossover operators, selection operators, and the method of sorting Pareto solutions in order to reduce energy consumption. With deadline restrictions, it also increases service profits. In order to improve the effectiveness and efficiency of computing, it starts by proposing a job scheduling architecture for cloud computing that includes a variety of components to analyze the application and allocate the proper resources to the applications.

The importance of task scheduling algorithms built on thorough QoS was highlighted by Hong Sun *et al*. [20]. The new Berger's Model is taken into account in the paper along with the benefit-fairness algorithm in a dynamic cloud computing environment. In the context of cloud computing, the New Berger game theory model applies game theory to task scheduling as well as social distribution theory. Task scheduling in the cloud is accomplished by allocating independent tasks to m virtual machine resources in order to fully utilize the resources with the shortest turnaround time.

Based on the cloud partitioning [21] concept, a better load balance model has been introduced for the public cloud with a switch mechanism to select various strategies for various circumstances. The algorithm enhances efficiency in the public cloud environment by applying game theory to the load balancing strategy. The cloud partitioning concept serves as the foundation for the load balancing strategy. When a job enters the system after the cloud partitions have been created, load balancing begins, with the main controller choosing which cloud partition should receive the job. After that, the partition load balancer chooses how to distribute the jobs among the nodes. This partitioning can be carried out locally when a cloud partition's load status is normal. This job needs to be moved to another partition if the cloud partition load status is abnormal. Every node's load data is collected by the cloud partition balancer, which uses it to assess the cloud partition's status. Table 1 shows how various criteria, including fairness, throughput, and waiting time, were used to evaluate the Load Balancing (LB) algorithms that were discussed.

Table 1: Strengths and Weaknesses of load balancing algorithms

Load balancing Algorithms	Strengths	Weaknesses
Static load balancing	At compile time, a choice is made regarding load balancing; Distributes the server's traffic equally; Less complex.	Restricted to environments with few variations in load; Lack the ability to deal with runtime changes in load.
Round Robin	Easy to comprehend; fairness; fixed time quantum; More effective during brief CPU bursts; Furthermore, priority (running time and arrival time).	Larger tasks take longer to complete; Due to the short quantum time, there may be more context switches; To achieve high performance, the job should be consistent.
Min- Min	Shortest time value for completion; It performs best when there are more minor tasks present.	Variation in the machines and tasks cannot be predicted; Starvation.
Max – Min	Preexisting requirements are known. Therefore, it functions better.	The task requires a lot of time to finish.
Dynamic load balancing	Fault tolerance; At runtime, distribute work; Only the system's current state is necessary.	Nodes must be continuously inspected; Considered to be more challenging.
Throttled load balancing	Decent performance; Tasks are managed using a list.	Tasks require waiting;
Honey Bee	Tends to decrease response time; Enhances throughput.	Without a VM machine, high priority tasks cannot be completed.
Ant - Colony	The ants can gather information more; Minimize make span; Perform independent tasks; Use computationally intensive methods.	Because of the overloaded network, searches take a long time; Uncertainty regarding the number of ants.
Genetic Algorithm	A higher level of resource utilization while meeting user expectations	Low convergence
Fuzzy logic based energy aware scheduling	Increase resource efficiency and meet user expectations	Less optimization
Particle Swarm Optimization (PSO)	Compared to conventional load balancing methods, it shortens the time required for VM migration; Reduces the downtime of the VMs, it improves Quality of Service (QOS).	Need to check more metrics; More complexity
Lion Optimization Algorithm	Provides significantly higher resource utilization than PSO and GA.	The results demonstrate that, despite a small difference, the cost is comparable to that of PSO and GA.

Conclusion

Load balancing, which entails spreading workloads equally among all nodes for greater resource usage and customer satisfaction, is a major difficulty in cloud computing. Several load-balancing methods for cloud computing are reviewed in this work. Round robin (RR), min-min, max-min, ant colony, genetic algorithm, honey bee, and other load balancing methods for cloud computing are all compared in this study. The comparison of several algorithms, taking into consideration aspects like fairness, throughput, fault tolerance, overhead, performance, response time, and resource usage, is a crucial part of this research. A drawback of earlier research is that each cloud computing technique does not take into account pertinent aspects like fairness, fast throughput, and equality.

The use of metaheuristic approaches for load balancing in cloud computing is thoroughly examined in this article. Particularly slow compared to evolutionary optimization approaches are metaheuristic methods. The obtained answers might not be exact ones. Therefore, a lot of research is being done to increase the degree of integration and effectiveness of the proposed solution. These issues have been investigated using metaheuristic approaches that reconfigure the transformation operator, extract features from the input workforce, and use a hybrid model. We also discuss several load-balancing strategies that are centered on a variety of performance factors. In the literature, a lot of academics have focused on significantly lowering end-to-end delay and performance costs.

The issues that need to be resolved were explored in order to create the best load balancing algorithms. This research also explored the advantages and disadvantages of various algorithms before contrasting them in light of the issues covered before. This examination and analysis of the literature has led us to the conclusion that while several algorithms may assess various variables, no single technique can enhance all load balancing measures. This study may serve as an inspiration for other researchers in the field to develop new algorithms that improve upon the present ones. Future work will concentrate on fixing the aforementioned problem and applying a hybrid strategy to raise system performance and security.

References

1. Katyal, Mayanka, Atul Mishra. "A comparative study of load balancing algorithms in cloud computing environment." arXiv preprint arXiv, 2014;1403:6918.
2. Xu M, Tian W, Buyya R. A survey on load balancing algorithms for virtual machines placement in cloud computing. Concurrency and Computation: Practice and Experience. 2017;29(12):e4123.
3. Dikaiakos MD, Katsaros D, Mehra P, Pallis G, Vakali A. Cloud computing: Distributed internet computing for IT and scientific research. IEEE Internet computing. 2009;13(5):10-3.
4. Srinivasan RK, Suma V, Nedu V. An enhanced load balancing technique for efficient load distribution in cloud-based IT industries. InIntelligent Informatics: Proceedings of the International Symposium on Intelligent Informatics ISI'12 Held at August 4-5 2012, Chennai, India, 2013, 479-485.
5. Tadi AA, Aghajanloo Z. Load Balancing in Cloud Computing using Cuckoo Optimization Algorithm. J. Innov. Res. Eng. Sci, 2018, 4(4).

6. Mishra SK, Sahoo B, Parida PP. Load balancing in cloud computing: a big picture. Journal of King Saud University-Computer and Information Sciences. 2020;32(2):149-58.
7. Buyya R, Ranjan R, Calheiros RN. Intercloud: Utility-oriented federation of cloud computing environments for scaling of application services. In Algorithms and Architectures for Parallel Processing: 10th International Conference, ICA3PP 2010, Busan, Korea, May 21-23, 2010. Proceedings. Part I 10, 2010, 13-31.
8. uzZaman SK, Maqsood T, Ali M, Bilal K, Madani SA, Khan AU. A load balanced task scheduling heuristic for large-scale computing systems. Comput. Syst. Sci. Eng. 2019;34:4.
9. Armstrong R, Hensgen D, Kidd T. The relative performance of various mapping algorithms is independent of sizable variances in run-time predictions. InProceedings Seventh Heterogeneous Computing Workshop (HCW'98), 1998, 79-87. IEEE.
10. R Lee, and B Jeng, Load-balancing tactics in the cloud`, In proceeding of International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), IEEE, (pp. 447-454).
11. Dasgupta K, Mandal B, Dutta P, Mandal JK, Dam S. A genetic algorithm (ga) based load balancing strategy for cloud computing. Procedia Technology. 2013;10:340-7.
12. Dam S, Mandal G, Dasgupta K, Dutta P. An ant colony based load balancing strategy in cloud computing. InAdvanced Computing, Networking and Informatics-Wireless Networks and Security Proceedings of the Second International Conference on Advanced Computing, Networking and Informatics (ICACNI. 2014;2:403-413). Springer International Publishing.
13. Mondal B, Dasgupta K, Dutta P. Load balancing in cloud computing using stochastic hill climbing-a soft computing approach. Procedia Technology. 2012;4:783-9.
14. LD DB, Krishna PV. Honey bee behavior inspired load balancing of tasks in cloud computing environments. Applied soft computing. 2013;13(5):2292-303.
15. Ebadifard F, Babamir SM. A PSO-based task scheduling algorithm improved using a load-balancing technique for the cloud computing environment. Concurrency and Computation: Practice and Experience. 2018;30(12):e4368.
16. Ebadifard F, Babamir SM. Dynamic task scheduling in cloud computing based on Naïve Bayesian classifier. Department of Computer Engineering University of Kashan, 2017.
17. Lakra AV, Yadav DK. Multi-objective tasks scheduling algorithm for cloud computing throughput optimization. Procedia Computer Science. 2015;48:107-13.
18. Liu J, Luo XG, Zhang XM, Zhang F, Li BN. Job scheduling model for cloud computing based on multi-objective genetic algorithm. International Journal of Computer Science Issues (IJCSI). 2013;10(1):134.
19. Sun H, Chen SP, Jin C, Guo K. Research and simulation of task scheduling algorithm in cloud computing. TELKOMNIKA Indonesian Journal of Electrical Engineering. 2013;11(11):6664-72.
20. Xu G, Pang J, Fu X. A load balancing model based on cloud partitioning for the public cloud. Tsinghua Science and Technology. 2013;18(1):34-9.
21. Sefati S, Mousavinasab M, ZarehFarkhady R. Load balancing in cloud computing environment using the Grey wolf optimization algorithm based on the reliability: performance evaluation. J Supercomput 2022;78(1):18–42
22. Singh RM, Awasthi LK, Sikka G. Towards metaheuristic scheduling techniques in cloud and fog: an extensive taxonomic review. ACM Computing Surveys (CSUR). 2022;55(3):1-43.
23. Lilhore U, Kumar S. Modified fuzzy logic and advance particle swarm optimization model for cloud computing. International Journal of Modern Trends in Engineering and Research (IJMTER). 2016;3(8):230-5.
24. Hu C, Deng Y, Min G, Huang P, Qin X. QoS promotion in energy-efficient datacenters through peak load scheduling. IEEE Transactions on Cloud Computing. 2018;9(2):777-92.