A Comparative Study Of Load Balancing Algorithms With Various Performance Metrics For Cloud Computing

Manpreet Kaur (Research Scholar)\textsuperscript{1}, Dr. Sarpreet Singh\textsuperscript{2}

\textsuperscript{1}Department of Computer Science, Sri Guru Granth Sahib World University, Fatehgarh Sahib (140406), Punjab, India.  
\textsuperscript{*}Corresponding Author

\textsuperscript{2}Department of Computer Science, Sri Guru Granth Sahib World University, Fatehgarh Sahib (140406), Punjab, India

Abstract
Cloud computing has received huge attention from researchers in the IT sector, business intelligence professionals and clients. It is basically an internet-based computing system with a sophisticated computational framework that provide services to clients globally and has various advantages over other computing systems. This is not a location-based computing model wherein centralized servers offer all information, programs, and resources to computers and other essential devices such as mobile phones, PDAs, and so on. In this paper, a brief introduction about cloud computing, its history is provided, and also discusses some major challenges that are faced in cloud computing. Moreover, this paper also studied the need for balancing the loads in cloud computing. In addition to this, the authors reviewed and studied different methods including meta-heuristic algorithms which include Ant colony optimization (ACO), Honey bee foraging, genetic algorithm or GA, hill climbing etc. that can be utilized for load balancing. Later on, a literature survey is also conducted in which more than 30 articles about cloud computing are analyzed, followed by some of the important load balancing factors. The study concludes that in order to improve the cloud services various factors such as processing time, complexity, scalability etc., plays an important role and while developing future systems much factors should be considered. This could help the systems to work for dynamic environments.

Keywords: Cloud computing, Load balancing algorithms, Task scheduling.

I. INTRODUCTION
During 1960s John McCarthy introduced the concept of Cloud computing, according to him “someday computing be structured as a public utility”. Later in the year 1966, Douglas Parkhill discussed the features of cloud computing in his book for the very first time. The word “Cloud” is basically related to the telecommunication industry, however different service providers started to provide Virtual Private Network (VPN) services that belong to same quality with low prices. Before the VPN services, the telecom providers used to provide end to end data circuits that resulted in
Although, the network was balanced by utilizing VPN services to transfer traffic. Cloud computing now encompasses servers as well as network infrastructure. Several industrial participants have embraced cloud computing and introduced it [1]. For instance, the Salesforce Company in 1999 through a user-friendly website began delivering applications to customers. Also, in the year 2002 the Amazon established AWS (Amazon Web Services) to provide computing and storage services [2]. Then, Amazon in 2006 released the Elastic Compute Cloud (EC2), a web service that was commercial and enabled small businesses and persons to lease computers to operate their software [3]. Furthermore, in 2008 to deploy private clouds the Eucalyptus released the very first open-source AWI API compatible platform [4]. In 2009, major companies like Google, HP, Microsoft, and Oracle started offering cloud computing services [5]. Then in 2010 Azure was launched by Microsoft. In 2012 Oracle cloud was introduced by Oracle that offered three business basics SAAS (Software-as-a-Service), PaaS (Platform-as-a-Service), and IaaS (Infrastructure-as-a-Service) [6]. It is clear from the above cloud’s evolution and history that after 2000 the cloud technology is growing rapidly and gaining the attention of several new customers and vendors.

Cloud computing is a modern computing platform which is used by each person now-a-days. The rapid growth of cloud computing is aided by the increasing number of computing technologies that are being developed at a reasonable cost of capacity and infrastructure. Cloud computing entails the subcontracting of computing assets with features such as redundant asset adaptability, on-demand provisioning, and low costs. Scalable assets are provided through dynamism as a service in cloud computing to ensure that economic benefits are spread across cloud users [7]. In addition, cloud computing provides features such as on-demand services, utility-based models, ease of use and accessibility, and a pay-per-use model, among others. IaaS, PaaS, and SaaS are basic types of cloud computing service models [8]. The software vendors in SaaS offer software like Google Drive and Gmail. In PaaS, a platform is given to write programming code, such as Google App Engine. In IaaS to build a platform similar to the cloud server, hardware resources are suggested for the user. Moreover, cloud computing services allow users to store and access files that are stored in several locations [9].

Cloud computing’s key concept is to offer similar services based on consumer demands by exchanging hardware resources and software programs. In general, CC can be categorized into three categories, those are, public, private, hybrid and group clouds. In the public cloud, service providers make services accessible over the Internet [10]. While as, private clouds are considered as center for data architectures that are operated either by a corporation or single government and offer functions like availability, scalability, provisioning, automation, and surveillance. In hybrid clouds, the services of private cloud and public clouds are incorporated to offer businesses more flexibility. [11] In a shared non-public cloud, a community cloud enables various organizations to share costs. It is a part of the public cloud that is distributed over a specific region and structured as a community. In cloud computing, one of the joint responsibilities of various service providers of the cloud and Enterprise’s IT department is security. Hence, when IT technology can be transferred to the cloud, but the CSP cannot fully take responsibility for information security [12]. Nowadays a static storage system is
inefficient because data gets lost when the storage location is unavailable for some purpose. Distributed storage networks comprise of many separate locations in computers linked through a private network or Internet, are being utilized to prevent such issues. Moreover, there seems to be no forced data duplication in these kinds of systems. As a result, if one computer is disconnected, data is unavailable [13].

II. CHALLENGES IN CLOUD COMPUTING

The cloud also faces various challenges like resource allocation, load balancing, data availability and management, interoperability, security, efficiency, privacy, compatibility, etc. The data center resources encounter challenges during the optimization of provisioning virtual networks while raising revenue. The difficulties in load balancing are dependent on storage utilization and download performance [11]. Other than these problems, there are many other problems which include migration of virtual machines, its security, QoS and resource use and so on in order to find out the optimum feasible solution for improving cloud resource utilization.

In cloud computing, the migration of virtual machines is considered a challenging task because they have different configurations and are not dependent. During the system overloading, few Virtual Machines migrate to a remote site through load-balancing mechanism and is called as VM migration. Since Cloud computing is a service-on-demand model, resources should be made available when a service request is made. The resources (usually virtual machines) must be transferred from one server to another server regularly, particularly at a different location. In such instances, load-balancing algorithm designers must address two issues, firstly, migrating time which has a direct impact on performance and secondly, security concerns or we can say the likelihood of attacks [14]. It is feasible to map the infrastructure in cloud computing environments and determine where the virtual machine is located. It is then possible to create a new Virtual Machine until one is in co-residence with the target VM. The VM attacker can access sensitive data from the legal VM attacked once it has been instantiated. This is a type of side-channel attack [15]. Because of the application’s dynamic workload and sharing of network resources by data centers, it is difficult for large scale data to guarantee the service level agreement (SLA) in cloud services. This allows elastic resources sealing which adjusts the VMs dynamically in order to meet the resource demands of application [16]. Furthermore, it allows VM movement [17] for load balancing [18] to enhance the resource usage and energy efficient applications.

The centers of cloud data are considered as one of the lively and uncertain resource usage patterns of users who are continually requesting virtual machines, alter their usage, unsteadiness in data arrival and departure of users and also the execution of hosts while dealing with variety of load levels may change. The performance of the cloud data center usually gets degraded when there are unbalanced loads which in turn leads to SLA breach and hence demands the need for utilizing the load balancing system.
The increased workload of VMs may cause one or more physical machine resources like, CPU, input output devices, memory usage and network bandwidth to become overwhelmed [19]. An overcrowded physical computer reduces the performing efficiency of all the virtual machines present in it, thus enhancing the data processing task, completion times and interactive application response times. This increased load should be shifted from overburdened computer to the less used computer so that such hotspots can be eliminated. However, load balancing via VM migration is a difficult and time-consuming challenge [20]. For starters, it takes into account many resources which include, CPU, memory space, I/O devices and so on for every VM and physical device [21]. Since, the resource requirements are different for every application, which means that they are operated by the VMs like CPU-intensive, memory-intensive, or network-intensive. While deciding which physical device or machine, a virtual machine should adopt to, the multidimensionality of virtual machine resource requirements must be assessed and leveled with multidimensional resources that are available on the physical machines [22].

III. MOTIVATION & NEED FOR BALANCING LOADS

Load balancing is a technique that maximizes the resources of cloud virtual machine. The basic objective of load balancing is to distribute resources and tasks evenly among all machines such that no node is overwhelmed or idle. The load balancing reduces waiting and response time increases resource utilization improves reliability, increases throughput outputs, and also enhances the performance of the system by managing each node [23]. Load balancers in high-availability servers may often also duplicate the user's session, which is required by the application. Instead of using only one algorithm for load balancing, using several algorithms and frameworks could improve reliability and performance. This paper provides a review of different load balancing methods with their advantages and disadvantages. In the next section of this paper, a literature survey on different load balancing schemes along with their proposed techniques for the VM migration and it’s deployment in cloud computing is given. Furthermore, various load balancing performance factors with detailed description are described. Then, a comparison analysis of various load balancing techniques in terms of their performances is done. And finally, a discussion with a conclusion and the future directions of Load Balancing and VM’s migration in Cloud computing is provided.

IV. LOAD BALANCING TECHNIQUES

In the context of Cloud computing, load balancing varies from conventional load balancing structure and application in that it makes use of the servers of the data center to process requests in a first-come, first-served manner [24]. The conventional algorithm of load balancing distributed requests based on the client's incoming requests. The techniques used for balancing the loads in Cloud computing can be categorized into three categories, that are given below:

- **Sender-Initiated Load Balancing**: The sender initiates this method of load balancing algorithm. The sender (user) will continue to send request messages until it discovers a recipient (machine) who can receives the load.
• **Receiver Initiated**: The receiver (machine) initiates the load balancing method in this case. The receiver will continue to send message request signals until it discovers a sender (user) who can get the load from it (machine).

• **Symmetric load balancing**: It this case the communication is initiated by both i.e. sender and receiver. On the basis of the present state of system, load balancing algorithms can be further be divided into two types, those are; static and dynamic balancing algorithms [25]. Fig 1 represents the flow chart of general load balancing algorithms.

![Flowchart of the Load Balancing Algorithms](image)

**Fig1. Flowchart of the Load Balancing Algorithms**

The static algorithms are best suited to a homogeneous and stable environment, however, when characteristics change frequently, the outcomes are poor. Following are some static load-balancing algorithm:

a) **Round Robin (RR)**: in this algorithm time slicing method is being used. Work is done in terms of rounds where every node is being assigned a time slot that have to wait for their turn. This time is separated and each node is given an interval to complete their tasks [26]. Since it
is impossible to predict the exact execution time in a cloud computing environment, static algorithms are not recommended. As a result, dynamic round-robin distinction has been suggested as a more effective way to deal with this problem [27]. Furthermore, the RR algorithm has been used in a variety of settings. It's used in CPU scheduling to divide CPU time among the tasks. Furthermore, it is used in the Cloud Computing environment to increase system speed and QoS (quality of service) supplied to users. Round Robin algorithm has two levels of application: at first it plans the CPU time in the virtual machine (VM) among the activities, and then it allocates resources for tasks. As shown in fig 2, the tasks would spread across the VMs in a round robin algorithm.

**Fig 2: Round Robin Algorithm Strategy**

Following is the pseudo code for the round robin Algorithm:

1. **Step 1:** At first the scheduler is initially moved to the ready queue.
2. **Step 2:** The data center controller decides which VM will receive the first request.
3. **Step 3:** At first the request is allocated randomly to any VM.
4. **Step 4:** VMs are arranged in a cyclic order after the first request is allocated.
5. **Step 5:** All VMs are transferred back to the VM that received the first request.
6. **Step 6:** In a cyclic order, the next request is allocated to the next VM.
7. **Step 7:** Repeat steps 3 for each request until the scheduler has finished processing.

b) **Opportunistic load-balancing algorithm:** The aim of this method is to keep every node occupied. As a result, the workload of each machine is ignored. OLB distributes uncompleted tasks to present available nodes at random, regardless of their workload. Each node's jobs are allocated in random order. It creates a load balance schedule, but it has an inadequate makespan. The OLB doesn't measure a node's execution time; the task is executed slower than normal causing bottlenecks although certain nodes are free [28].

c) **Min-min load-balancing algorithm:** The method starts with an unassigned series of tasks. First, the task's minimum completion period is determined. Then a minimum value is selected from among them that would be the shortest time for all tasks on any asset. The task is configured on the appropriate machine as per the required minimum time on the machine,
the execution time for all the other tasks is changed, and the task is taken off the list. This method is tracked until all tasks have been allocated to a resource. This algorithm outperforms other algorithms when the lot of small tasks is bigger than the number of large tasks. Moreover, one disadvantage of this strategy is that it would lead to starvation [29]. The pseudo code for Min-min load balancing algorithm is given below:

Step 1 for all tasks ti in MT
Step 2: for all machines mj
Step 3: CTij = ETij + rj
Step 4: Perform until all of the tasks in MT have been mapped
Step 5: for each task ti in MT
Step 6: Locate minimum CTij and then obtain resources.
Step 7: Locate task tk that has minimum CTij.
Step 8: Allocate tk to resource ml that obtains it
Step 9: Remove tk from MT
Step 10: Upgrade drj
Step 11: Update CTij for all i
Step 12: End

Where, CTij stands for completion, ETij stands for expected execution time of task ith on resource jth, and rj stands for the ready time for resource jth.

d) **Max min load-balancing algorithm**: it is quite identical to the min load balancing algorithm, the only difference is that after determining the lowest execution times, the highest value is selected that is the total time spent by all tasks upon the resources. The task is then configured on the corresponding computer based on the highest time. All tasks have their execution times changed, and delegated tasks have been excluded from the tasks list that has been allocated to the computers. Since all of the specifications are known ahead of time, this algorithm works flawlessly [30]. When multiple allocation tasks arrive in the same batch, take the following steps:
Step 1: At first select the task that has maximum execution time
Step 2: Using the VM database, calculate the anticipated time taken in every task in each VM.
Step 3: Select the VM with the lowest completion time (Min).
Step 4: At last, assign the task to the appropriate virtual machine. Furthermore, update VM's operations their total execution time in the status table of VM.

e) **Two-phase load-balancing algorithm**: In this method, OLB and Min-Min scheduling algorithms are incorporated to create more efficient and low-maintenance load balancing systems. To achieve load balancing, OLB holds each node operational, whereas the reason for using the Min-Min scheduling algorithm is to reduce the processing time for every task on the
neighboring node, reducing the total execution time. As a result, hybrid solutions aids in the effective use of resources and increases productivity [31].

f) **Throttled Load Balancer:** In this method a table is constructed, which contains the VMs as well as current state. The generated request is submitted to the data center's control unit when a VM is allocated to execute a specified task. The data center tries to find the finest virtual machine to match their strengths to the required task. If a suitable VM is not available, the load balancer will transmit -1 back to the data center [32]. Throttled Load Balancer is demonstrated in Fig 3.

![Throttled Load Balancer Algorithm](image_url)

**Fig 3: Throttled load Balancer Algorithm**

g) **Active Monitoring Load Balancing (AMLB) Algorithm:** in this method the record of each and every VM present workload is kept along with the requests that are assigned to it. The least loaded VM is identified, if a request for new VM is received. When there are several requests, the first one found is chosen. The particular id of the load balancer is then sent to the data center controller which transmits this request to the virtual machine along with the id assigned to it. This notifies the current VM load balancer that the new allocation is done. When tasks are allocated, its processing power is ignored and only the present load of the VM is considered. This results to, some job waiting times may rise, infringing on the QoS requirement [33]. The AMLB algorithm is depicted in fig 4.
The dynamic load balancing algorithm distributes the workload across processors during runtime. Dynamic algorithms, unlike static algorithms, assign processes dynamically only when any processor is overburdened. Following are the list of some dynamic load balancing algorithms.

h) **Honey bee foraging Algorithm:** It can be defined as the decentralized load balancing approach that is inspired by nature and focused on honey bee action. This method aids in load balancing through heterogeneous cloud nodes. The node’s current load is determined first, and then the algorithm determines if the node is underloaded, overloaded, or balanced [34]. Following is the pseudo code for the Honey bee foraging algorithm is:

i=0 generates the initial population.
Evaluate the starting population's Fitness Quotient and sort the population depending on the fitness result.

While i Fitness Quotient i-1
   Step 1: i = i + 1
   Step 2: Chose the right patches for your neighborhood search from both elite and non-elite patches.
   Step 3: Drive forager bees to the best patches, both elite and non-elite.
   Step 4: Compute each patch's Fitness Quotient.
   Step 5: Filter the findings by fitness.
   Step 6: Allocate the remaining bees to the non-ideal areas for global search.
   Step 7: Determine the Fitness Quotient of patches that aren't the best.
   Step 8: Filter the findings by their fitness
   Step 9: Continue running until the termination criteria are met.
   Step 10: FINISH

**Fig 4: Active Monitoring Load Balancing [8]**
i) **Active clustering Algorithm:** It's an advanced version of random sampling and uses the clustering principle. The key concept of this method is to cluster related nodes and function with these nodes. A matchmaker node is a node that starts a procedure and then chooses the other process. A matchmaker node establishes a link between an initial node and its neighboring node and then disconnects it. This procedure is repeated iteratively for balancing the load [35].

j) **Ant Colony Optimization Algorithm based on Load balancing:** The existence of real ants in search for food creates a network, inspired this algorithm. Whenever a request is made, the ant moves forward, visiting each node one by one, and records the result of overloaded nodes or underloaded. If the ant encounters an overloaded node, it begins moving backward to the previous underloaded node to exchange data [36]. Assume that the transition probability value $P_{ij}^k(t)$ of ant k from resource nodes i to j at time t, utilizes this value will be used to choose the next job to schedule as represented in equation (1)

$$P_{ij}^k(t) = \begin{cases} \tau_{ij}(t)^\alpha * [n_{ij}(t)]^\beta / \sum_{i=0}^{n}[\tau_{ij}(t)]^\alpha * [n_{ij}(t)]^\beta, & j \in A \\ 0, & \text{Others} \end{cases} \quad (1)$$

Where $\tau_{ij}$ is considered as the pheromone value to transmit resources from i node to j node at time t, $\alpha$ is considered as the heuristic factor that denotes the relative merits of the order in which events are scheduled and $\beta$ is considered as the expect heuristic factor that in the ant selection scheduling sequence, exemplifies the significance of heuristic knowledge.

k) **Stochastic Hill Climbing Algorithm:** This algorithm is considered solves the optimization problem. This strategy is divided into two types namely, total and fragmented. This algorithm assures an explicit response in two different ways: by ensuring that no such task exists and by locating a likely significant task to the variable. Deficient process, does not guarantee correct responses for the majority of the information sources. SHC is considered local optimization algorithmic programs, which move in an upward direction to increase the value. This algorithm stops working if no neighbor fuses a higher value a fitness function (F) is crucial to analyze a candidate solution to execute the SHC method efficiently. After removing the immunized nodes, which is indicated by S (solution), the fitness problem is described as the residual networks biggest connected component size [37]. LCC calculates the network's vulnerability to a worst-case epidemic size as illustrated in equation (2).

$$F(S) = LCC(G/\{i|s_i = 1\}) \quad (2)$$

Step 1: At first a network G is considered, k is considered as the immunization resources, and $t$ is the number of time steps to stop.
Step 2: Then randomly a candidate solution (S) is generated from solution space, then repeat select I as the random bit from the immunized groups (|i|S_i = 1)).
Step 3: Select v as a random bit from non-immunized groups (|i|S_i = 0))
Step 4: ΔF = if the values of u and v are exchanged, the difference in F
Step 5: If ΔF<=0 replace values of u and v in S
For t iterations, F must not change.

1) **Genetic Load Balancing Algorithm:** this method executes in a powerful cloud environment and employs a careful registering method. In contrast to the rule of static algorithms, this algorithm allows for faster execution. The benefit of the rule is that it is well-managed a massive inquiry home, applicable to cutting-edge objective work, and will not be tacked into native ideal solution [38]. GA's have a three-stage execution process.

   a) **Selection Operator:** in this step, initial population is chosen randomly.
   b) **Crossover Operator:** in this step for crossover, locate a fitness pair of people.
   c) **Mutation Operator:** Mutation value is a modest or low probability value. These bits are switched between 0s and 1s. The result is a new group of people who are ready to cross over.

m) **Generalized Priority Algorithm:** This algorithm works by prioritizing tasks based on their size, prioritizing default machines depending upon the processor power, and then selecting the suitable VM for the priority of the needed task [39]. The pseudo code of this algorithm is described below:

   Pseudo code:
   
   Step 1: Make a virtual machine list for each Datacenter.
   Step 2: Find processing speed, memory, cost processor, and storage of sewer (VM)
   Step 3: **End** loop
   Step 4 Find array length and then sort cloudlet array where computational power is considered important
   Step 5: **For** each virtual machine fund index in LB table Mime VM allocation=0
   Step 6: Fill array cloudlet when the length of array = MIPS count
   Step 7 **For** suitable cloudlet find virtual machine.
   Step 8 Search a matched VM id
   Step 9: **End** loop
   Step 10: **End** loop
   Step 11 Update the list of datacenter and virtual machine.

Table 1: Different algorithms with their nature, advantage and disadvantage

<table>
<thead>
<tr>
<th>Algorithm of Load Balancing</th>
<th>Dynamic or Static in nature</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Round Robin Algorithm</td>
<td>Static</td>
<td>This algorithm has fixed time interval and has a better outcome for short</td>
<td>Bigger tasks take longer to complete.</td>
</tr>
<tr>
<td>Algorithm</td>
<td>Static</td>
<td>CPU burst and when the no. of operations is greater than processors this algorithm acts well</td>
<td>For task, completion takes a longer time</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>------------</td>
<td>--------------------------------------------------------------------------------------------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td>Max-Min Algorithm</td>
<td>Static</td>
<td>It is more effective because the requirements are understood in advance. It performs better that Min-Min algorithm and in this algorithm small tasks are greater in no.</td>
<td></td>
</tr>
<tr>
<td>Min-Min Algorithm</td>
<td>Static</td>
<td>It provides the best outcomes for small tasks and enhance the lifespan</td>
<td>Starvation</td>
</tr>
<tr>
<td>Opportunistic Load Balancing</td>
<td>Static</td>
<td>It enhances the performance as well as utilization of resources</td>
<td>For task, completion takes a longer time</td>
</tr>
<tr>
<td>Weighted Round-Robin Load</td>
<td>Static</td>
<td>It takes into account the VMs' resource abilities and provides a better amount of work to the improved resource VMs based on their weightage.</td>
<td>The length of the tasks while selecting the suitable VM is not done.</td>
</tr>
<tr>
<td>Throttled Load Balancer:</td>
<td>Static</td>
<td>This algorithm improves resource utilization and significantly reduces the average execution time</td>
<td>This algorithm has not been simulated in a specific workload environment and in this algorithm do not find time limit</td>
</tr>
<tr>
<td>Algorithm</td>
<td>Type</td>
<td>Description</td>
<td>Additional Information</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>--------</td>
<td>----------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Ant Colony Optimization Algorithm</td>
<td>Dynamic</td>
<td>It is computationally fast and minimizes the lifespan</td>
<td>Takes a long time for searching</td>
</tr>
<tr>
<td>Honey Bee Foraging</td>
<td>Dynamic</td>
<td>High throughput and less response time</td>
<td>Low priority mode takes a lot of time</td>
</tr>
<tr>
<td>Active clustering Algorithm</td>
<td>Dynamic</td>
<td>It is considered a self-aggression algorithm that groups similar nodes together.</td>
<td>This algorithm performs low in terms to Honeybee foraging algorithm when total nodes are increased.</td>
</tr>
<tr>
<td>AMLB</td>
<td>Static</td>
<td>It discovers the least loaded VM, improves performance, and each time it updates the index table when a task is completed.</td>
<td>This algorithm takes a lot of time to evaluate index table and it provides better outcome when the variation is low during workload</td>
</tr>
<tr>
<td>Stochastic Hill Climbing Algorithm</td>
<td>Dynamic</td>
<td>This algorithm tackles bottleneck issues, provides efficient system workload distribution</td>
<td>This algorithm do not have any effective way to tackle optimization issues</td>
</tr>
<tr>
<td>Genetic Load Balancing Algorithm:</td>
<td>Dynamic</td>
<td>This algorithm improves the system’s efficiency, shortens the timing of task and improves resource utilization.</td>
<td>Provides small throughputs, lack of scalability, and does not save power.</td>
</tr>
<tr>
<td>Generalized Priority Algorithm</td>
<td>Dynamic</td>
<td>This algorithm prioritizes tasks based on their size and the machine's computing power.</td>
<td>Distribute all tasks across all accessible VM.</td>
</tr>
</tbody>
</table>

**OTHER RELATED WORKS**

In a Cloud Computing network, the load balancer is considered a crucial part because it enhances the environment of Cloud Computing, uses several tasks to accomplish user’s requirements, and gives
users the ability to uniformly spread the load. The main motive for a load-balancing algorithm is to improve the resource usage, improve the system’s performance, and to provide better response time [40]. Without the help of load balancing, novel spinning virtual servers would not be able to take incoming traffic in a unified manner. Load balancing can also detect unreachable servers and divert traffic to others servers which are operating at the backend [41]. Various researchers have proposed various techniques in the domain of Load balancing in cloud computing, some of them are discussed here: R. M. Alguliyeva et al. (2019) [42], to resolve the issue of scheduling tasks in Cloud Computing, the authors of this paper developed a PSO based αPSO-TBLB (Task-Based Load Balancing) load balancing technique. Marwa Gamal et al. (2019) [43]: the researchers of this paper suggested numerous algorithms for achieving effective load balancing, including the hybrid metaheuristic algorithms methodology that incorporates the osmotic behavior with bio-inspired methods of load balancing L. Kong et al. 2019 [44]: To effectively solve the NP-problem, the author in this paper proposed a fast heuristic algorithm that was based on the zero-imbalance approach. This method mainly focused on reducing the completion time difference in heterogeneous virtual machines without using any priority methods or complex scheduling decisions, that frequently subject heuristic algorithms to the specific cloud configuration. Divya and Bijendra 2019 [45]: The load scheduling methodologies necessitate a large number of resources and non-adaptive, stable procedures in computations, resulting in increased response and waiting time, and also increases the total cost of computing. To address this, the authors of this paper presented the Hybrid Genetic-Gravitational Search Algorithm (HG-GSA) load scheduling methodology. Karan D. Patel et al. 2019 [46], to balance the workload on a cloud-based system the authors of this paper offered a load balancing technique that combines two strategies. For priority-based activities, authors employed an upgraded honey bee behavior-inspired algorithm, and for non-priority-based tasks, authors utilized a modified weighted round-robin method. L. Yu, et al. 2020 [47]: Since the demand of resources and workload of VM’s were highly dynamic, these schemes made inefficient load balancing migrations. To tackle this problem, the researchers developed a stochastic load balancing scheme that provides a likelihood guarantee for resource overloading while minimizing total migration overhead. Soumen Swarnakar et al. 2020 [48], the authors of this paper developed a novel scheme for all incoming requests across VM’s by using IDLBA (Improved Dynamic Load Balancing Approach).

Virtualization is used in cloud computing to multiplex and partition the physical infrastructure of the machine [49]. Configuration of VM determines how physical resources are shared among VMs, for security reasons it keeps on segregating from other VMs [50]. A PM (Physical Machine) or a server must supply all of the facilities that the VM demands, like memory, storage network bandwidth and a CPU [51]. The software layer called VMM (VM Monitor) manages the VMs inside a PM or a server as represented in fig. 5.
Fig 5: Hosted Virtualization

In general, the VM monitor allows you to create, migrate, and terminate VMs [52]. In cloud computing virtual machine is considered an intriguing feature that aims to respond to the demands of dynamic VM to ensure that cloud clients receive the guaranteed SLA (Service Level Agreement). As a result, when a VM demands resources that the hosted PM could not give, the VM is transferred to another PM to fulfill the VM's request. Additionally, VMs may be transferred to improve the administration of physical equipment and data centers. The most significant procedure carried out in the area of VM migration is VM placement that selects the most appropriate PM for the virtual machine. Static VM placement refers to a mapping of VM that remains constant throughout. The dynamic VM placement is defined as allowing the placement to adapt in response to the system's load. Proactive and reactive VM placements are two types of dynamic VM placements. The modifications to the original placement are typically performed only once the system enters an undesirable condition are done in the reactive VM placement. Modifications to initial placement are permitted in proactive VM Placement before the system approaches a particular condition [53]. Several researchers have proposed various techniques for locating the VMs in the cloud, so that an effective solution is achieved, some of them are discussed here:

- **Ashwin and B. Annappa, (2019) [54]**: the authors of this paper tackled the problem of VM placement optimization in heterogeneous data centers, and proposed an effective context-aware adaptive heuristic-based approach that saves energy and enhances data center performance efficiency.

- **Sambit et al. (2018) [55]**: a complete mapping algorithm was proposed by the researchers of this paper to address the problem of VM placements.

- **Nimisha and Hiren, (2017) [56]**: the author in this paper developed a method, namely, Host Utilization Aware (HUA) in which the underutilized host is detected and its VMs are shifted to another host dynamically.

- **Minxian Xu, et al. (2017) [57]**: the researchers of this paper focused on various load balancing methods for placing VMs in cloud data centers, and the studied methods were then categorized as per tehri classification.

- **H. Tian, et al. (2017) [58]**: The scholars of this paper examined the problem's NP-hardness and presented heuristic techniques to minimize the total execution time of the PMs in both offline and online scenarios.

- **Abdelaziz A., et al. (2017) [59]**: To discover the best VMs in a cloud network, the author created a novel intelligent architecture for HCS and utilized three intelligent algorithms which were, GA, PSO, and parallel particle swarm optimization or PPSO.

- **R. Shea, et al. 2014, [60]**: the author in this paper addressed both Hardware Virtualization and Paravirtualization
add considerable energy overhead that affected both sending and receiving web-server. Gu, et al. (2014) [61]: The author concentrated on open research concerns such as VM service billing, power budgeting, and energy-saving schedules in this study. To address this issue, the author examined their efficiency as well as current power metering methods. J. Dong, et al. (2013) [62]: The authors of this paper addressed the VM placement problem by designing a novel greedy algorithm in which minimum cut with the best-fit is combined and also proposed a VM placement method that meets multiple resource constraints. K. Li, et al. (2013) [63]: This work introduces a novel NP-hard problem that can be reduced to a knapsack problem. As a result, the author developed an emulated VM migration approach for off-line VM installation. Furthermore, the researchers investigated a hybrid system in which a batch was used to accept future virtual machines for the on-line scenario. O. Biranet et al., (2012) [64]: Min Cut Ratio-aware VM Placement is a new optimization problem developed by the author (MCRVMP). The author used many heuristics related to both placement computation and run-time performance to get good results for medium-sized data centers in a reasonable amount of time.

V. LOAD BALANCING PERFORMANCE FACTORS

To put it in a more practical sense, the cloud LB parameters would not only improve the processed output of LB but will also speed up the process and will also lay the theoretical groundwork for investigating efficient LB algorithms in cloud computing load. Load balancing is measured by different performance factors that are classified into two classes one is the parameters with quantitative attributes and another one is qualitative attributes. The details of parameters under both categories are given discussed below:

a) **Response Time:** this parameter helps to evaluate load balancing algorithm's performance. The period between the commencement of a procedure and its conclusion is known as response time. Processes must be dispersed evenly among the nodes by utilizing proper load balancing techniques. Good reaction time also implies that processes in the system do not have to wait for long period [65]. The following formula can be used to calculate response time:

\[
T = F - A \quad \text{-------}(3)
\]

Where, T represents the response time, and F represents the duration of the current CPU burst. S is the current CPU burst's start time.

b) **Scalability:** A device can perform consumer functions inside the dynamic flow of traffic. The LB algorithm should scale up during peak periods and downscale during off-peak periods. This reflects the probability of a functioning program surviving an increase in the size or volume of workload. Algorithm's fault tolerance power is unaffected by the number of nodes in the process [66].

c) **Fault Tolerance:** The system capability to run reliably in the event of a system failure, resulting in increased resilience and availability. A fault-tolerant LB method would ensure
that network losses are kept to a minimum owing to network overload. This parameter demonstrates the algorithm's capacity to handle fault scenarios and also has capability to recover from failures [67].

d) **Throughput**: While performing load balancing, this parameter determines the total activities completed in a unit of time. It also specifies the level where a LB algorithm is used to execute a computation. The goal of this parameter is to improve performance. Tasks that have been completed within a specific time frame and the maximum number of dead per unit of time [68].

e) **Migration Time**: The time required to transmit operations through unbalanced devices is known as the transfer period. In other words, it can also be defined as the time taken to transfer overcrowded VMs from one Physical Machine to another Physical Machine, as in the VM transmits Load Balancing [66].

f) **Resource Utilization Factor**: It can be defined as the services that is available for the overall accessible resources. This defines how much a virtual machine uses the tools. When a VM becomes overburdened, the tasks use a significant amount of energy, which is an undesirable event because the tasks cannot be completed rapidly. Greater resource utilization implies reduced resources that translate to less free services. As a result, a good LB algorithm generates the most of the available resources. [69].

g) **Power Saving**: In this parameter, the metric specifies the amount of power and strength consumed by the VM following the LB procedure. In a virtual machine, an efficient LB algorithm reduces power and energy consumption. [66].

**Comparison of several metrics of Load Balancing**

This section shows the essential load balancing metrics that have been studied in previous methodologies with scheduling based on eight main characteristics i.e., throughput, scalability, fault tolerance, makespan (time taken by a set of jobs for completion), response time, execution time, resource utilization, and task rejection ratio. Table 2 illustrates different LB approaches that are compared based on the specified parameters.

**Table 2: Different LB approaches are compared based on specified parameters.**

<table>
<thead>
<tr>
<th>S.N</th>
<th>References</th>
<th>Algorithm Used</th>
<th>Publication year</th>
<th>Throughput</th>
<th>Scalability</th>
<th>Fault Tolerance</th>
<th>Makespan</th>
<th>Response Time</th>
<th>Execution Time</th>
<th>Resource Utilization</th>
<th>Task rejection Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[70]</td>
<td>Conventional non classical</td>
<td>2017</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Algorithm [70]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Classical and Linear Programming</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2018</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

|   | Dragonfly optimization and constraint measure based load balancing          |
|   | 2017                                                                        |
|   | ✓                                                                           |

|   | GA and Min-Min                                                              |
|   | 2016                                                                        |
|   | ✓                                                                           |
|   | ✓                                                                           |
|   | ✓                                                                           |

|   | BFO+ Lamarack Evolutionary                                                  |
|   | 2017                                                                        |
|   | ✓                                                                           |
|   | ✓                                                                           |

|   | Fairness Aware Algorithm                                                    |
|   | 2017                                                                        |
|   | ✓                                                                           |
|   | ✓                                                                           |

|   | GA                                                                          |
|   | 2017                                                                        |
|   | ✓                                                                           |

|   | ACO and PSO                                                                 |
|   | 2015                                                                        |
|   | ✓                                                                           |
|   | ✓                                                                           |
|   | ✓                                                                           |

|   | GA and GEL                                                                  |
|   | 2015                                                                        |
|   | ✓                                                                           |
|   | ✓                                                                           |

|   | Agent based Nature Inspired Algorithm                                       |
|   | 2015                                                                        |
|   | ✓                                                                           |
|   | ✓                                                                           |
|   | ✓                                                                           |

|   | Honey Bee Algorithm                                                         |
|   | 2016                                                                        |
|   | ✓                                                                           |
|   | ✓                                                                           |
|   | ✓                                                                           |

|   | Non-Classical                                                              |
|   | 2015                                                                        |
|   | ✓                                                                           |
|   | ✓                                                                           |
|   | ✓                                                                           |
VI. CONCLUSION

The word “cloud” in cloud computing addresses to a cluster of networks that is basically an extension of grid, distributed and parallel computing. In cloud computing, the person doesn’t have restricted access to cloud computing paradigms at any time. Typically, the consumers opt a middleman that provides internet connectivity in Cloud Computing rather than establishing their personal network architecture. The consumers should only ever pay for services they have been using. In this review article, a comprehensive and brief summary about the cloud computing, its evolution and challenges are discussed. Moreover, we have also analyzed that due to the rising number of users on internet, there is a need for balancing loads in order to provide cloud services effectively to the user. For this, a literature survey is conducted in which more than 30 different literatures that were proposed by various researchers are analyzed and discussed. After reviewing these literatures, it was observed that optimization algorithms like PSO, GA, honey bee foraging etc. were able to balance loads efficiently but still has a scope of improvement. In addition to this, the performance of cloud computing models, can be enhanced by considering important Quality of service (QoS) attributes like make span, stability of system, execution time and so on.

REFERENCES


http://www.webology.org
56. Nimisha Patel, Hiren Patel, "Energy efficient strategy for placement of virtual machines selected from underloaded servers in compute Cloud", Journal of King Saud University - Computer and Information Sciences, 2017