ASTES

Advances in Science, Technology and Engineering Systems Journal
Vol. 6, No. 2, 860-870 (2021)
www.astesj.com
Special Issue on Multidisciplinary Sciences and Engineering

ASTES Journal
ISSN: 2415-6698

Load Balancing Techniques in Cloud Computing: Extensive Review

Ahmad AA Alkhatib[*], Abeer Alsabbagh, Randa Maraqa, Shadi Alzubi

Alzaytoonah University of Jordan, Amman, 11183, Jordan

ARTICLE INFO

Article history:
Received: 21 December, 2020
Accepted: 24 March, 2021
Online: 04 April, 2021

Keywords:
Cloud computing
Load Balancing
Software Defined Network
Static Algorithms
Dynamic Algorithm

ABSTRACT

It has become difficult to handle traditional networks because of extensive network developments and an increase in the number of network users, and also because of new technologies like cloud computing and big data. Traditional networks are experiencing an increase in VM load and in the time taken for processing tasks. Hence, it has become essential to modify the traditional network architecture. A notion called Load balancing techniques that increases the conformance of network management was presented recently to deal with this problem. The critical need for load balancing emerges due to network resources limitations and requirements fulfillment that facilitates traffic distribution through various resources to enhance the efficiency and reliability of network resources. This task has been carried out by several researchers before, who have presented various algorithms with their benefits and shortcomings. The focus of this research is on the notion of cloud computing load balancing and on the advantages and disadvantages of a chosen load balancing algorithm. Furthermore, it examines the metrics and issues of these algorithms.

1 Introduction

Cloud Computing has universally greater interest in web technologies currently. With the increasing demands of the cloud, popular website's servers are getting overloaded, in order to fulfill users requirement load balancing is one of the promising solutions. Load balancing is the procedure of sharing the load between multiple processors in a distributed environment to minimize the turnaround time taken by the servers to cater service requests and make better utilization of the available resources. cloud computing incorporates and enhances the advantages of a few existing distributed computing technologies. Network computing refers to a "model of distributed computing that employs geographically distant resources and hence, allows users to gain access to computers and data and to handle their accounts from varied locations". Virtualization is another technology that hides the physical properties of computing resources to obscure complexity when dealing with applications, systems or end-users [1]–[3].

Cloud computing initially represented online business applications, referred to as application service provision (ASP). Later on the term was used more extensively following the distribution of services by the company, where the name given to each service provider was based on the service they offered to the customer [4].

Since its introduction in late 2006, several definitions of cloud computing have been formulated. It can essentially be defined as a framework for allowing on-demand network access to a shared set of online configurable computing resources (like network, storage, server, services and applications) [5],[6].

The definition of cloud computing provided by HP is as follows: "Everything as a Service" [7]. On the other hand, Microsoft considers cloud computing to signify "Cloud and the Client" [8]. According to T-Systems, cloud computing refers to "renting infrastructure, software and bandwidths under specified service conditions. It should be possible to modify these components routinely on the basis of the customer requirements and should be widely available and secure. Furthermore, there are 2-end service level agreements (SLAs) and use-dependent service invoices that are part of cloud computing" [9].

It is possible to distinguish cloud computing into three models, i.e. SaaS (SOFTWARE–AS-A SERVICE), which provides a user interface to the user which they can access through a browser or desktop application; PaaS (PLATFORM-AS-A SERVICE, which offers applications development tools and a programming language execution environment to the user; and finally IaaS (INFRASTRUCTURE- AS-A-SERVICE), which offers virtual computerized resources to the user that they can manage in accordance with their requirements. These models can also be classified in accordance with the services they offer [10], [11].

- Database as a Service (DaaS)
- Storage as a Service (SaaS)
- Network as a Service (NaaS)

[*]Corresponding Author: Ahmad AA Alkhatib, Ahmad.Alkhatib@zuj.edu.jo

- Expert as a Service (EaaS)
- Communication as a Service (CaaS)
- Security as a Service (SECaaS)
- Monitoring as a Service (Maas)
- Testing as a Service (TaaS)

The Cloud Computing categories are shown in Figure 1.

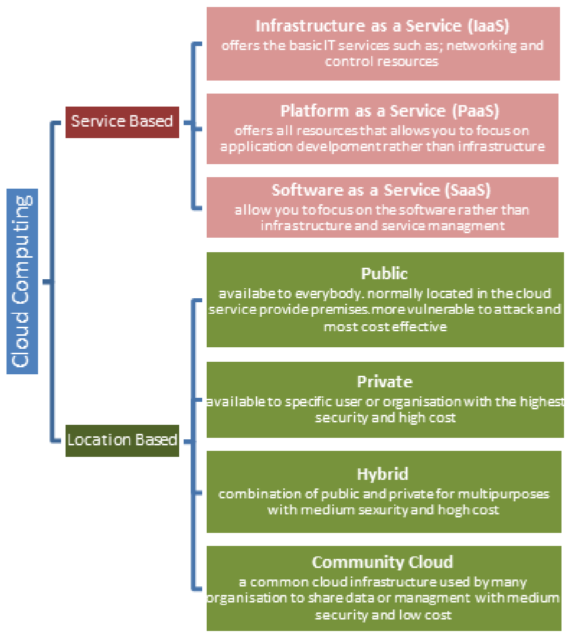

Cloud Computing
Service Based
Infrastructure as a Service (IaaS)
offers the basic IT services such as; networking and control resources
Platform as a Service (PaaS)
offers all resources that allows you to focus on application develpoment rather than infrastructure
Software as a Service (SaaS)
allow you to focus on the software rather than infrastructure and service managment
Location Based
Public
availabe to everybody. normally located in the cloud service provide premises. more vulnerable to attack and most cost effective
Private
available to specific user or organisation with the highest security and high cost
Hybrid
combination of public and private for multipurposes with medium sexurity and hogh cost
Community Cloud
a common cloud infrastructure used by many organisation to share data or managment with medium security and low cost

Figure 1: Cloud Computing categories

Cloud computing is becoming quite popular; hence, there has been a significant increase in the amount of processing being carried out in the clouds. However, cloud computing is experiencing several issues in providing the required services [11], including:

- Load Balancing
- Performance analysis and modeling
- Throughput and response time
- Security and privacy
- Resource management
- QoS

The structure of the rest of the paper is as follows: the Load Balancing Concept and Challenges described in Section 2. Section 3 presents the load balancing techniques classification . A few static and dynamic load balancing algorithms are discussed in Section 4. Section 5 Discussion and comparison with the latest studies in this field are compared and Lastly,section 6 gives a conclusion of the study .

2 Load Balancing

There are a series of nodes in the cloud that are connected to one another. Here, one of the nodes is either selected randomly or on the basis of an algorithm to fulfill requests made by users. There may be different data and hence, there is different load on every node, and each node in a cloud can have unequal load of tasks based on the quantity of work requested by clients. Load balancing is a very significant issue that made it imperative for cloud computing to distribute load on the resources available so as to achieve lower response time. This makes sure that no VMs in the system are overloaded at any time [12]. In addition, balance workload continues to be an issue in the cloud as they are unable to determine the quantity of demand that is obtained in the cloud [13].

Load balancing concepts face several challenges because there are various physical and logical issues that may have an impact on the technique being used. A few of these issues are listed in Table I [11]:

Table 1: LOAD BALANCING ISSUES

Graphical Distribution node	There is geographical distribution of data centers in clouds. On the other hand, distributed nodes are considered as a single system without taking into account various factors like communication delay, networking delay and distances between nodes, resources and users.
Virtual Machine Migration	Multiple VM are allowed by the VM concept on the same Physical Machine. Due to the distinct VM structure, the physical machine may become overloaded.
Algorithm Complexity	To ensure that it does not affect the efficiency of the cloud, load balancing algorithm should be simple and concise.
Heterogeneous nodes	In these times, users have different requirements. It is essential to have heterogeneous nodes to fulfill users' need for services. The load balancing decision is influenced by the heterogeneity of the nodes.
Single point of Failure	In general, load balancing algorithm is typically carried out on a central node to assign tasks. The entire computing fails in case the central node fails.
Load Balancer Scalability	Computing power, topology, storage, etc. determine the response time of load balancing

3 Load Balancing Techniques Classification

The classification of the algorithms depends on the existing status of the system and involves two categories [14], [11]: static algorithms that obtain information pertaining to the system and identify the existing resources for using before commencing. The load is distributed on the existing VM till the work concludes, which is preferably used when the VM capabilities are similar to one another. This type of algorithm has various techniques, for example round robin, Min-Min, weight, Max-Min, honey bee foraging, throttled, etc. The issue faced with static algorithms is that they depend on static information which is unable to effectively demonstrate dynamic load variations taking place within the VMs [15]. Dynamic algorithms are the second kind of algorithms and are different from static algorithms in the sense that they exhibit flexibility. These algorithms work on the basis of rescheduling the tasks allocated to them over the available VM while carrying out the work. Better performance is exhibited by static algorithms in contrast to the dynamic algorithm, whereas with respect to the competitive ratio, dynamic algorithms have a larger ratio compared to the static algorithm [16]. The typical structure of load balancing in cloud environments is demonstrated in Figure 2, whereas Figure 3 shows the classification of the load balancing methods. Several parameters are used to assess load balancing algorithms, which are regulated by various policies. The metrics are presented in Table 3 [15], [17], while an outline of the load balancing policies is given in Table 4.

Table 2: Load Balancing Metrics

Metrics	Description
Performance	It refers to ensuring the consistency and efficiency of the algorithms carried out
Response time	It refers to the time taken by the system to give a reaction to the user request. he more rapid the response time, the higher the satisfaction obtained by the user.
Throughput	The number of tasks carried out for each unit of time.
Scalability	The ability of the algorithm to handle the tasks of the user in an efficient and effective way.
Fault Tolerance	The robustness of the algorithm to solve issues and errors so that its performance can be regained
Migration Time	The time needed to shift the load from a certain VM to another in accordance with the VM loading during situations of overload or under-load.
Resource Usage	Determines the optimum use of computing resources in the data center that should be employed by the algorithm
Makespan	Time needed to carry out and treat a given set of tasks

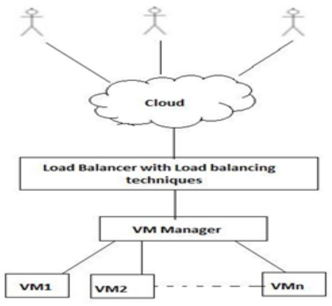
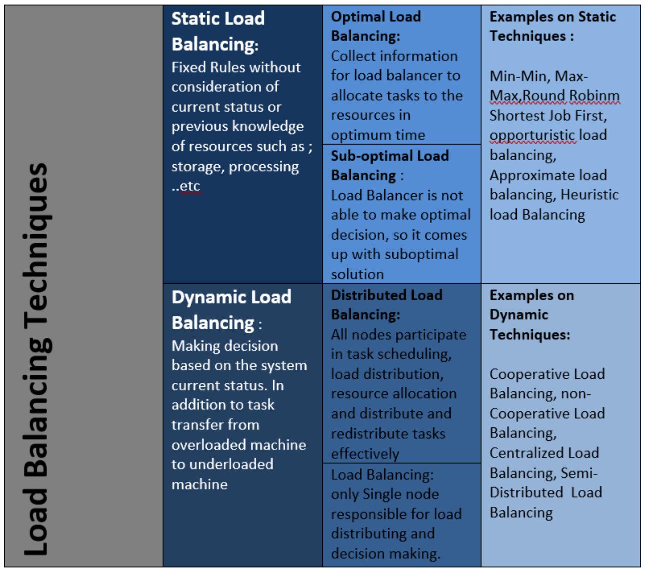


Figure 2: General Structure of Load balancing in Cloud Environment [18]



Load Balancing Techniques			
	Static Load Balancing: Fixed Rules without consideration of current status or previous knowledge of resources such as ; storage, processing ..etc	**Optimal Load Balancing:** Collect information for load balancer to allocate tasks to the resources in optimum time	**Examples on Static Techniques :** Min-Min, Max-Max,Round Robinm Shortest Job First, opporturistic load balancing, Approximate load balancing, Heuristic load Balancing
		Sub-optimal Load Balancing : Load Balancer is not able to make optimal decision, so it comes up with suboptimal solution	
	Dynamic Load Balancing : Making decision based on the system current status. In addition to task transfer from overloaded machine to underloaded machine	**Distributed Load Balancing:** All nodes participate in task scheduling, load distribution, resource allocation and distribute and redistribute tasks effectively	**Examples on Dynamic Techniques:** Cooperative Load Balancing, non-Cooperative Load Balancing, Centralized Load Balancing, Semi-Distributed Load Balancing
		Load Balancing: only Single node responsible for load distributing and decision making.	

Figure 3: Load balancing techniques classification

Table 3: Load Balancing Policies

Selection Policy	The required tasks that need to be transferred from a certain VM to another are specified in this policy. This is done depending on the extent of overhead that is to be transferred
Location policy	In this policy, the tasks are sent to the free, under-loaded and available VMs so that they can be fulfilled. The desired VM is determined in this policy on the basis of the availability of the required services so that the task can be transferred in accordance with the available technique, like Negotiation, Random and Probing.
Transfer policy	In this policy, the conditions for shifting the task from a local VM to another local or remote VM are determined. Two types of tasks are used in this regard: the existing tasks and the last task received.
Information policy	This policy deals with keeping information pertaining to the resources safe in the system to ensure that other policies can benefit from them while making decisions.

4 Load Balancing Common Algorithms

A few static and dynamic load balancing algorithms will be presented in this section, with benefits and drawbacks for each algorithm:

4.1 *Round Robin Algorithm*

This static algorithm is one of the simplest methods used following the selection of the existing VMs. One of these VMs is randomly chosen by the data center unit to commence its operations. In addition, it is organized in a circular manner. After this, every VM that gets a request is shifted towards the final part of the list [19], [20][21],.

There is, however, no interaction between this algorithm and the distinct abilities of the VMs because the tasks are equally divided on the VMs, even though they have distinct abilities (Figure 4) [15]. Hence, Weight Round Robin algorithm is used to enhance this algorithm, which assigns weight for each VM on the basis of its abilities, and then offers the ability to each node to have a specific number of tasks on the basis of the weights allocated to each VM. This algorithm is considered similar to Round Robin algorithm from the perspective of time division because it distributes the time as a circular. However, the distinction is that the tasks are offered to VM on the basis of particular restrictions, for example checking weights [22], [23].

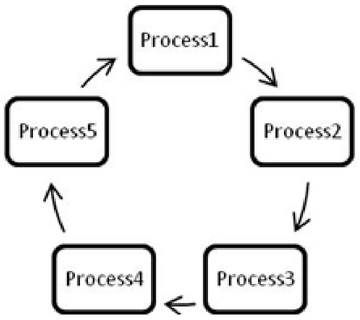


Figure 4: Round Robin Load Balancer [19]

4.2 Throttled Load Balancer (TLB)

A table is generated in this algorithm that includes the virtual machines as well as the existing state (available/busy). If a specific task is allocated to a virtual machine, a request is made to the control unit within the data center, which will look for the ideal VM suit with respect to their abilities to achieve the required task [24]. The load balancer will send -1 back to the data center if an appropriate VM is not available [15]. Figure 5 presents a demonstration of Throttled Load Balancer.

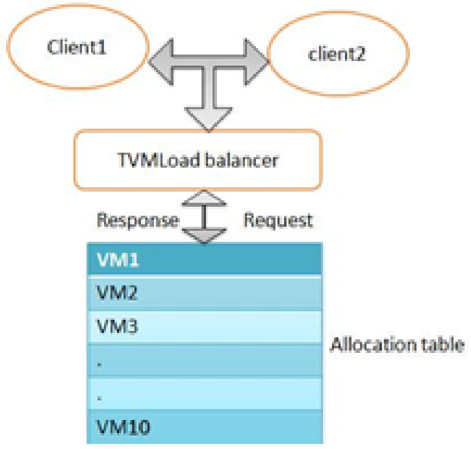


Figure 5: Throttled Load Balancer

In contrast, the process of looking for the ideal virtual machines always takes place from the start of the table each time; therefore, certain VMs are not employed. A modified throttled algorithm was put forward by [25], which works by changing the process of VM selection. After receiving the subsequent request, the VM at index adjacent to the VM already allocated is selected based on the state of the VM. In addition, [22] presented an efficient throttled algorithm, which consisted of 3 algorithms, i.e. throttled algorithm, ESCE (Equally Spread Current Execution algorithm) and Round Robin. This algorithm is considered as an advancement of the main throttled algorithm, where the enhanced algorithm employed the data structure to maintain information regarding the VMs. In contrast, Hash Map index is used to search for VM and allocated tasks where it works at a faster pace compared to the throttled algorithm.

Furthermore, Divide-and-Conquer and Throttled algorithm (DCBT) approach is taken to be the hybrid approach because it gives preference to the order when allocating the VM. This is done to achieve the highest usage of resources by reducing the overall time taken to execute the task. In addition, this algorithm seeks to update the table by dividing the independent jobs equally among all the VMs. Hence, it works faster than the throttled algorithm [26].

4.3 Min-Min Load Balancing Algorithm

This algorithm is easy to use and works at a faster pace [27]. In addition, it improves performance and consists of a series of tasks. The time taken to execute the task is computed and allocated to VMs on the basis of the smallest completion time for the existing tasks. The process will continue till it is ensured that each task has been allocated to VM [16]. Because of the existence of a greater number of smaller tasks, this algorithm performs better compared to if there were bigger tasks. However, this will lead to starvation because of giving priority to smaller tasks and deferring the bigger tasks [21].

4.4 Max Min Load Balancing Algorithm

As stated by [28], [29], this algorithm is quite similar to the Min-Min Load Balancing, based on the calculation time. In this algorithm, all existing tasks are sent to the system, after which the calculation is carried out for determining the least time to complete each of the given tasks. The selected task then has the maximum time to be completed will be allocated to the relevant machine [11]. A comparison of the performance of this algorithm with the Min-Min algorithm shows that the Max-Min algorithm is better because there is just one large task in the set, which means that the Max-Min algorithm will carry out the shorter tasks alongside the larger task [15].

4.5 Opportunistic Load Balancing Algorithm

This algorithm is a type of static algorithm that is not capable of describing the existing workload of the VM; therefore, it allocates the tasks randomly to all nodes in the system to ensure they are all working [30], [31], and [32]. Tasks are accomplished at a slow pace through this algorithm as it does not compute the existing implementation time [33]. Hence, it provides incorrect outcomes for the load balance [15].

4.6 Ant Colony Optimization

This algorithm that is based on ACO allows for distributing the workload efficiently between the nodes of a cloud. The working of this algorithm is influenced by the Ant concept, where it looks for a new pathway if it comes across an obstacle and allocates a new route between the nodes [34]. The algorithm functions by first initializing the table, carrying out the data flow and achieving the threshold level for the required nodes. When the flow passes through the nodes, the algorithm examines the node; if it is under load, then it uses the highest trailing pheromone (TP) that provides the route for the under-loaded node. The table is then updated till the node achieves the threshold limit. Nonetheless, if the threshold limit is attained, it uses the Foraging Pheromone (FP) to examine new sources of food and update the table till it attains an under-load, after which it reassigns resources. This cycle keeps on occurring till the completion of the process.

4.7 Honey Bee Algorithm

The working of this algorithm is influenced by the way bees behave when looking for honey, because its main objective is to divide the workload on the VM, considering the lack of excessive resource utilization and lack of under-usage of resources. This algorithm works by choosing a VM that fulfills two main requirements. Fewer tasks are allocated to this VM compared to those assigned to other VM machines. The time taken by VM to process falls within the average processing time taken by all other VMs [35], [23].

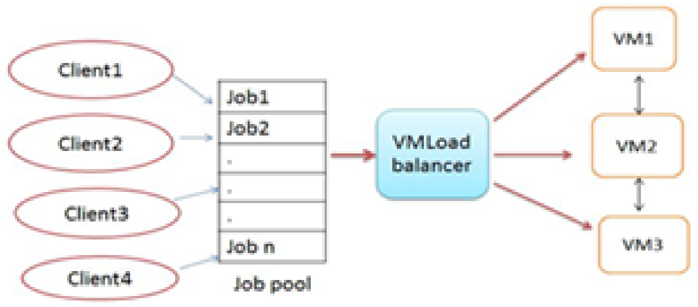


Figure 6: Active Monitoring Load Balancerr

4.8 Active Monitoring Load Balancer (AMLB)

It is a type of dynamic load technology [21]. This technology obtains information relevant to each VM and to the number of requests that are presently allocated to each of them [24]. The Data Center Controller (DCC) scans the VM index table after receiving a new request to determine the VM that is least loaded or idle. First-come-first serve concept is employed by this algorithm to allocate load to the VM that has the smallest index number for more than two servers [21].

The VM ID is sent back by the AMLB algorithm to the DCC which then sends the request to the VM represented by that ID. The AMLB is informed about the new allocation by the DCC and it is sent the cloudlet [24].

Once the task is completed, the information is sent to the DCC and the VM index table is reduced. When a new request is received, it goes over the table again using load balancer and then the process allocation occurs. Figure 6 presents an illustration of AMLB.

4.9 Genetic Algorithm

In the process of genetic algorithms, scheduling takes place on the basis of the biological notion of population generation. The purpose of using this algorithm was to enhance the distribution of load within the cloud, where the algorithm starts functioning using the preliminary population procedure. The initial population consists of the set of all individuals who are involved in determining the optimal solution. Each solution in the population is referred to as an individual, and each individual is described as a chromosome to make it appropriate for carrying out genetic operations [36]. It was presumed by the authors in [37] that a comparison of the newly developed population will be carried out with the previous one. The solutions are then chosen to obtain solutions (offspring) on the basis of its fitness function [38]. The fitness function is used to determine the quality of individuals in the population with respect to the specified optimization goal [37].

For every chromosome, the fitness function is computed, after which the most appropriate results are chosen to be used as parents. The operation then commences with a crossover process; a part of each parent is used to create a new child; this child is then improved by employing a mutation process. This process keeps on occurring till the best results are obtained [39]. The basis of genetic algorithms is randomness; however, it is not the same as the random search in that it approves the most suitable individuals within a population. The crossover rate and mutation probability values have a significant impact on the performance of the genetic algorithm [38].

The general scheme of the genetic algorithm is shown in Figure 7.

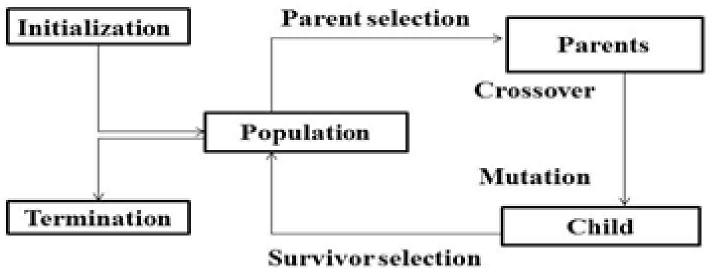


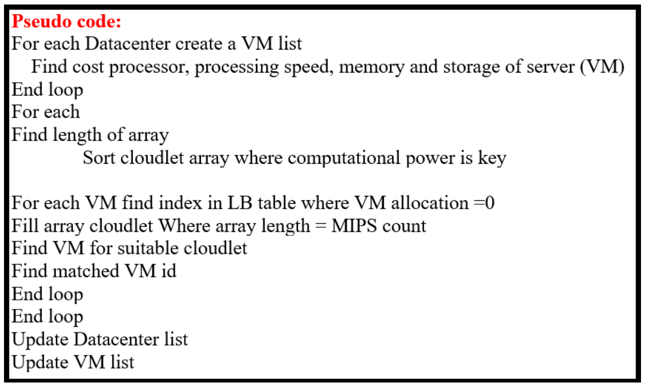
Figure 7: General scheme of genetic algorithm [40]

4.10 First Come First Serve

This algorithm works by distributing new tasks to resources that have the least waiting time (i.e. resources that have the least number of tasks). Here, there is sequential execution of the tasks, with the work commencing from the first task till its completion, after which the subsequent task from the queue is carried out. This algorithm is used as it allocates tasks to the virtual machines without considering the specifications of the virtual machines available and also the time that will be taken by the tasks in queue when allocating new tasks [41].

4.11 *Generalized Priority Algorithm*

The mechanism of action of this algorithm is to give priority to the tasks according to their size (Higher missions size take higher priority), and the default machines are given priority based on the power of the processor, then the appropriate virtual machine is chosen for the priority of the required task [41]. An example on GPA suppose you have 6 VMs represented by their Id and processor speed VMs = 0, 850 , 1, 1000, [5], 3, 150, 4,250, 5,750. Here the priority goes to VM 4 will because it has the highest speed, then priority is given to VM 2 and then VM 6 and so on [41]. This algorithm works as in Figure (pseudo):



```
Pseudo code:
For each Datacenter create a VM list
  Find cost processor, processing speed, memory and storage of server (VM)
End loop
For each
Find length of array
        Sort cloudlet array where computational power is key

For each VM find index in LB table where VM allocation =0
Fill array cloudlet Where array length = MIPS count
Find VM for suitable cloudlet
Find matched VM id
End loop
End loop
Update Datacenter list
Update VM list
```

Figure 8: K. Generalized Priority Algorithm Pseudo code

4.12 *NBST Algorithm*

Initially, this algorithm assumes that the number of instructions required for the tasks is on the waiting list and their length is known, then it arranges all available virtual devices according to their implementation speed MIPS (Million instructions per second) and arranging the cloudlets according to their length. Where this algorithm depends on dividing the number of VM and cloudlets into two parts after arranging them in descending order until access to the VM or cloudlets one at most, then assigning the VM groups to the cloudlets groups [42].

4.13 *Load Balancing Technique based on Cuckoo Search and Firefly*

The cuckoo search is based on the brooding behavior of cuckoo bird, where this bird laid their eggs on other bird's nest, the best quality eggs are carried forward to the next generation and the worst nests are abandoned [43]. Cuckoo Search with Firefly is a hybrid algorithm is used for loud balancing in a cloud computing environment; this algorithm schedules the tasks and allocates the overloaded VMs tasks to the under-loaded VMs. This algorithm finds the best VM in less time and improves the loud balance efficiency and avoids the task imbalanced situations in the entire system [44]. The LB technique starts with scheduling the tasks using Round Robin method, then calculate the capacity and the load of each VM, then it identifies the overloaded and under-loaded tasks using Cuckoo Search with Firefly to start the migration of tasks from overloaded VMs to under-loaded VMs [44]. It avoids the task imbalanced situations in the entire system. Also this method has the advantages of high interchange rate and less number of tuning parameter. This method has the advantages of high interchange rate and less number of tuning parameter. The proposed method migrates 2 task, while existing method requires 7 task for migration.

4.14 *predicted load balancing algorithm based on the dynamic exponential smoothing*

This algorithm predicts the load of the VMs by using the dynamic exponential smoothing model. Where the dynamic balancing algorithms focus on the fastest response speed algorithm and the least connection algorithm [45].

Exponential smoothing is one of prediction algorithms based on time series, which takes advantage of all historical data, and differentiates them through the smoothing factor to allow recent data make a greater impact on the analytical value than long-term data, [46].

The allocated node has the shortest corresponding time, when a new request arrives to the server node that has a little connection numbers, the dynamic algorithm takes the load characteristics of the server node as the reference, makes the adjustment with every weighted factor, and reflects the real-time load under the support of weighted value, by that the algorithm helps to find the corresponding smoothing coefficient with the VM load time series of current phrase, and helps to make prediction with the load value at the next moment of this VM[45]. The load predicted balancing algorithm developed by the improvement that permits to construct the dynamic smoothing exponent and to obtain the load prediction with a higher accuracy through the comparison of short load time series. Also it helps to provide more accurate service demand for users and enhance the resource utilization ratio of the server node to a higher level [45]. The accuracy of the model prediction can be affected, because of the selection standard of α exponential smoothing is not very clear [45].

5 Discussion

A comparison of the load balancing algorithms is presented in this section. Each load balancing algorithm is described in specific identifiers with its main properties and limitations.

5.1 *Round Robin*

- *Algorithm_Name: [Round Robin].*
- *Algorithm_Type: [Static].*
- *Algorithm_Overhead: [No Overhead].*
- *Algorithm_Degree_of_complexity: [Simple].*
- *Algorithm_Strength_points :*
  - *Simplicity*
  - *Easy to install*
- *Algorithm_Limitations:*
  - *Cannot be improved any more*
  - *No multi-tasking capabilities*

  - *Could lead to over head*
  - *Does not take capacity importance and task duration into account*

- *Algorithm_Performance: [Low Performance].*
- *Algorithm_Throughput: [Low Throughput].*
- *Algorithm_Fault_tolerance: [No FT].*

5.2 Throttled

- *Algorithm_Name: [Throttled].*
- *Algorithm_Type: [Static].*
- *Algorithm_Overhead: [No Overhead].*
- *Algorithm_Degree_of_complexity: [Medium].*
- *Algorithm_Strength_points :*
  - *The data center searches for the best VM that fits its capabilities with the task required*
  - *Maximization of resource usage*
  - *Increase average of execution time*
- *Algorithm_Limitations:*
  - *Does not define time limits*
  - *Could result in idle VMs*
- *Algorithm_Performance: [medium Performance].*
- *Algorithm_Throughput: [high Throughput].*
- *Algorithm_Fault_tolerance: [FT].*

5.3 Min-Min

- *Algorithm_Name: [Min-Min].*
- *Algorithm_Type: [Static].*
- *Algorithm_Overhead: [Medium Overhead].*
- *Algorithm_Degree_of_complexity: [Simple].*
- *Algorithm_Strength_points :*
  - *Quick*
  - *Easy to install*
- *Algorithm_Limitations:*
  - *Leads to starvation*
- *Algorithm_Performance: [Increased performance for smaller tasks].*
- *Algorithm_Throughput: [Improved Throughput].*
- *Algorithm_Fault_tolerance: [No FT].*

5.4 Max-Min

- *Algorithm_Name: [Max-Min].*
- *Algorithm_Type: [Static].*
- *Algorithm_Overhead: [Medium Overhead].*
- *Algorithm_Degree_of_complexity: [Simple].*
- *Algorithm_Strength_points :*
  - *Execute the large tasks in parallel with short tasks*
- *Algorithm_Limitations:*
  - *Leads to starvation.*
- *Algorithm_Performance: [Better performance than min-min].*
- *Algorithm_Throughput: [Improved Throughput].*
- *Algorithm_Fault_tolerance: [No FT].*

5.5 Opportunistic

- *Algorithm_Name: [Opportunistic].*
- *Algorithm_Type: [Static].*
- *Algorithm_Overhead: [Low Overhead].*
- *Algorithm_Degree_of_complexity: [Simple].*
- *Algorithm_Strength_points :*
  - *The advantage is quite simple and reach load balance*
- *Algorithm_Limitations:*
  - *No fairness in task assignment*
  - *Tasks are planed slowly because it does not determine the nodes current execution time*
  - *the whole completion time (Make span) is very poo*
- *Algorithm_Performance: [poor Performance].*
- *Algorithm_Throughput: [Limited Throughput].*
- *Algorithm_Fault_tolerance: [No FT].*

5.6 Ant Colony Optimization

- *Algorithm_Name: [Ant Colony Optimization].*
- *Algorithm_Type: [Dynamic].*
- *Algorithm_Overhead: [High Overhead].*
- *Algorithm_Degree_of_complexity: [High complexity].*
- *Algorithm_Strength_points :*
  - *The style of living ants depends on the*
  - *transition from VM to another VM.*
  - *Better performance*
  - *Reduce operation time*
  - *Can be combined with heuristic search algorithm*
- *Algorithm_Limitations:*
  - *Complex and load on the system unit.*
  - *Untrusted in reality*
- *Algorithm_Performance: [High Performance].*
- *Algorithm_Throughput: [High Throughput].*
- *Algorithm_Fault_tolerance: [FT].*

5.7 Honey Bee

- *Algorithm_Name: [Honey Bee].*
- *Algorithm_Type: [Dynamic].*
- *Algorithm_Overhead: [High Overhead].*
- *Algorithm_Degree_of_complexity: [High complexity].*
- *Algorithm_Strength_points :*
  - *it is chosen the best VM by comparing the cost of executing a task on one VM with the other available VMs.*
  - *Takes into account excessive resources and under used ones*
  - *Number of assigned tasks less than number of handled by other VMs to reduce execution time*
- *Algorithm_Limitations:*
  - *Complexity*
  - *Increase the machine size does not increase throughput*
- *Algorithm_Performance: [High Performance].*
- *Algorithm_Throughput: [Low Throughput].*
- *Algorithm_Fault_tolerance: [FT].*

5.8 AMLB

- *Algorithm_Name: [AMLB].*
- *Algorithm_Type: [Static].*
- *Algorithm_Overhead: [High Overhead].*
- *Algorithm_Degree_of_complexity: [medium complexity].*
- *Algorithm_Strength_points :*
  - *It identifies the least loaded VM*
  - *Increase performance*
  - *Update index table every time task executed*
- *Algorithm_Limitations:*
  - *Time consuming in index table calculation*
  - *algorithm gives better results when there is low variation in workload*
- *Algorithm_Performance: [High Performance].*
- *Algorithm_Throughput: [High Throughput].*
- *Algorithm_Fault_tolerance: [FT].*

5.9 Genetic Algorithm

- *Algorithm_Name: [Genetic Algorithm].*
- *Algorithm_Type: [Dynamic].*
- *Algorithm_Overhead: [High Overhead].*
- *Algorithm_Degree_of_complexity: [High complexity].*
- *Algorithm_Strength_points :*
  - *Executed the cloudlets in less time*
  - *Improve load distribution in the cloud*
  - *Better resource usage*
  - *Better approve the best fitted population indexing in VMs*
- *Algorithm_Limitations:*
  - *Complexity*
  - *High resources required for population generation and calculation*
  - *Crossover mutation is highly effected by probability parameters*
- *Algorithm_Performance: [High Performance].*
- *Algorithm_Throughput: [High Throughput].*
- *Algorithm_Fault_tolerance: [FT].*

5.10 First come first serve

- *Algorithm_Name: [First come first serve].*
- *Algorithm_Type: [Dynamic].*
- *Algorithm_Overhead: [There is Overhead].*
- *Algorithm_Degree_of_complexity: [Simple complexity].*
- *Algorithm_Strength_points :*
  - *Distribute tasks to all available VMs*
- *Algorithm_Limitations:*
  - *Not preventive*
  - *assigning tasks to all virtual machines without paying attention to the specifications of the available virtual machine*
  - *not paying attention to the time that the tasks in the queue will take when distributing the new tasks*
- *Algorithm_Performance: [Medium Performance].*
- *Algorithm_Throughput: [Medium Throughput].*
- *Algorithm_Fault_tolerance: [No FT].*

5.11 Generalized Priority Algorithm

- *Algorithm_Name: [Generalized Priority Algorithm].*
- *Algorithm_Type: [Dynamic].*
- *Algorithm_Overhead: [No Overhead].*
- *Algorithm_Degree_of_complexity: [medium complexity].*
- *Algorithm_Strength_points :*
  - *give priority to the tasks according to their size and the machine processor power*
- *Algorithm_Limitations:*
  - *take more execution time*
  - *could lead to starvation*
- *Algorithm_Performance: [Increased Performance].*
- *Algorithm_Throughput: [High Throughput].*
- *Algorithm_Fault_tolerance: [FT].*

5.12 NBST Algorithm

- *Algorithm_Name: [NBST Algorithm].*
- *Algorithm_Type: [Dynamic].*
- *Algorithm_Overhead: [Yes the is Overhead].*
- *Algorithm_Degree_of_complexity: [medium complexity].*
- *Algorithm_Strength_points :*
  - *Distribute all task on all available VM*
- *Algorithm_Limitations:*
  - *take more time in an assign VMs for cloudlets*
- *Algorithm_Performance: [increased Performance].*
- *Algorithm_Throughput: [High Throughput].*
- *Algorithm_Fault_tolerance: [FT].*

5.13 Load Balancing based on CSF

- *Algorithm_Name: [Load Balancing based on CSF].*
- *Algorithm_Type: [Dynamic].*
- *Algorithm_Overhead: [medium Overhead].*
- *Algorithm_Degree_of_complexity: [medium complexity].*
- *Algorithm_Strength_points :*
  - *schedules the tasks and allocates the overloaded VMs tasks to the under-loaded VMs*
  - *avoids the task imbalanced situations in the entire system*
  - *high interchange rate and less number of tuning parameter*
- *Algorithm_Limitations:*
  - *multiple algorithms have to be followed*
- *Algorithm_Performance: [Improved Performance].*
- *Algorithm_Throughput: [High Throughput].*
- *Algorithm_Fault_tolerance: [FT].*

5.14 predicted load balancing algorithm based on the dynamic exponential smoothing

- *Algorithm_Name: [predicted load balancing algorithm based on the dynamic exponential smoothing].*
- *Algorithm_Type: [Dynamic].*
- *Algorithm_Overhead: [Yes there is Overhead].*
- *Algorithm_Degree_of_complexity: [High complexity].*

- *Algorithm_Strength_points :*
  - *provide more accurate service demand for users and improve the resource utilization of the VMs*
- *Algorithm_Limitations:*
  - *the accuracy of the model prediction can be affected, because of the selection standard of α exponential smoothing is not very clear*
- *Algorithm_Performance: [Improved Performance].*
- *Algorithm_Throughput: [Medium Throughput].*
- *Algorithm_Fault_tolerance: [No FT].*

6 Conclusion and Future Work

This paper presents an extensive Review for load balancing that carry out load distribution among the VM in a different way. considering the lack of excessive resource utilization and not keeping the VMs idle. As a service that is carried out over the network is known as cloud computing, where a lot of significance is given to load balancing issues. The performance will decrease with an overloaded system. Hence, smart load balancing algorithm is needed to maintain the position of QoS. a description for algorithm techniques has been explained and followed by a comparison between them.

Conflict of Interest The authors declare no conflict of interest.

Acknowledgment The Authors Thanks The Alzaytoonah University of Jordan for their support.

References

[1] A. A. AlKhatib, T. Sawalha, S. AlZu'bi, "Load Balancing Techniques in Software-Defined Cloud Computing: an overview," in 2020 Seventh International Conference on Software Defined Systems (SDS), 240–244, IEEE, 2020,doi: 10.1109/SDS49854.2020.9143874.

[2] N. A. Sultan, "Reaching for the "cloud": How SMEs can manage," International journal of information management, **31**(3), 272–278, 2011,doi: https://doi.org/10.1016/j.ijinfomgt.2010.08.001.

[3] S. AlZu'bi, Y. Jararweh, H. Al-Zoubi, M. Elbes, T. Kanan, B. Gupta, "Multi-orientation geometric medical volumes segmentation using 3d multiresolution analysis," Multimedia Tools and Applications, **78**(17), 24223–24248, 2019, doi:https://doi.org/10.1007/s11042-018-7003-4.

[4] A. Susarla, A. Barua, A. B. Whinston, "Understanding the 'service'component of application service provision: an empirical analysis of satisfaction with ASP services," in Information Systems Outsourcing, 481–521, Springer, 2006, doi: https://doi.org/10.1007/978-3-540-34877-1_17.

[5] G. Lu, W. H. Zeng, "Cloud computing survey," in Applied Mechanics and Materials, volume 530, 650–661, Trans Tech Publ, 2014, doi: DOI: 10.4028/www.scientific.net/AMM.530-531.650.

[6] A. Abusukhon, M. N. Anwar, Z. Mohammad, B. Alghannam, "A hybrid network security algorithm based on Diffie Hellman and Text-to-Image Encryption algorithm," Journal of Discrete Mathematical Sciences and Cryptography, **22**(1), 65–81, 2019, https://doi.org/10.1080/09720529.2019.1569821.

[7] S. Robison, "Everything-as-a-Service: A blue sky view of the cloud," 2009.

[8] L. Xin, C. Song, "Cloud-based innovation of Internet long tail," in 2011 International Conference on Product Innovation Management (ICPIM 2011), 603–607, IEEE, 2011, doi: https://doi.org/10.1007/s11042-019-7367-0.

[9] R. F. El-Gazzar, "A literature review on cloud computing adoption issues in enterprises," in International Working Conference on Transfer and Diffusion of IT, 214–242, Springer, 2014, doi: 10.1007/978-3-662-43459-8_14.

[10] V. Paul, S. Pandita, M. Randiva, "CLOUD COMPUTING REVIEW," 2018.

[11] P. Kumar, R. Kumar, "Issues and challenges of load balancing techniques in cloud computing: a survey," ACM Computing Surveys (CSUR), **51**(6), 1–35, 2019, doi:https://doi.org/10.1145/3281010.

[12] E. J. Ghomi, A. M. Rahmani, N. N. Qader, "Load-balancing algorithms in cloud computing: A survey," Journal of Network and Computer Applications, **88**, 50–71, 2017, doi: https://doi.org/10.1016/j.jnca.2017.04.007.

[13] M. R. Mesbahi, M. Hashemi, A. M. Rahmani, "Performance evaluation and analysis of load balancing algorithms in cloud computing environments," in 2016 Second International Conference on Web Research (ICWR), 145–151, IEEE, 2016, doi:10.1109/ICWR.2016.7498459.

[14] K. Al Nuaimi, N. Mohamed, M. Al Nuaimi, J. Al-Jaroodi, "A survey of load balancing in cloud computing: Challenges and algorithms," in 2012 second symposium on network cloud computing and applications, 137–142, IEEE, 2012, doi:10.1109/NCCA.2012.29.

[15] S. K. Upadhyay, A. Bhattacharya, S. Arya, T. Singh, "Load optimization in cloud computing using clustering: a survey," Int. Res. J. Eng. Technol, **5**(4), 2455–2459, 2018.

[16] L. Xu, M. Xu, J. Wang, R. Semmes, Q. Wang, H. Mu, S. Gui, H. Yu, W. Tian, R. Buyya, "A Data-driven Approach to Identify Resource Bottlenecks for Multiple Service Interactions in Cloud Computing Environments," .

[17] S. Afzal, G. Kavitha, "Load balancing in cloud computing–A hierarchical taxonomical classification," Journal of Cloud Computing, **8**(1), 22, 2019, doi: https://doi.org/10.1186/s13677-019-0146-7.

[18] M. A. Shahid, N. Islam, M. M. Alam, M. M. Su'ud, S. Musa, "A Comprehensive Study of Load Balancing Approaches in the Cloud Computing Environment and a Novel Fault Tolerance Approach," IEEE Access, **8**, 130500–130526, 2020, doi:10.1109/ACCESS.2020.3009184.

[19] R. Sajjan, B. R. Yashwantrao, "Load balancing and its algorithms in cloud computing: a survey," International Journal of Computer Sciences and Engineering, **5**(1), 95–100, 2017.

[20] M. M. D. Shah, M. Kariyani, M. Agrawal, "Allocation of virtual machines in cloud computing using load balancing algorithm," International Journal of Computer Science and Information Technology & Security (IJCSITS), **3**(1), 2249–9555, 2013.

[21] N. R. Tadapaneni, "A Survey Of Various Load Balancing Algorithms In Cloud Computing," 2020.

[22] G. Megharaj, K. Mohan, "A survey on load balancing techniques in cloud computing," IOSR Journal of Computer Engineering (IOSR-JCE), **18**(2), 55–61, 2016, doi: 10.9790/0661-18215561.

[23] V. Mathur, "A Comparative Study of Load Balancing Techniques in Distributed Systems," .

[24] J. James, B. Verma, "Efficient VM load balancing algorithm for a cloud computing environment," International Journal on Computer Science and Engineering, **4**(9), 1658, 2012.

[25] S. G. Domanal, G. R. M. Reddy, "Load balancing in cloud computing using modified throttled algorithm," in 2013 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), 1–5, IEEE, 2013, doi: 10.1109/CCEM.2013.6684434.

[26] S. G. Domanal, G. R. M. Reddy, "Load balancing in cloud environment using a novel hybrid scheduling algorithm," in 2015 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), 37–42, IEEE, 2015, doi : 10.1109/CCEM.2015.31.

[27] V. Arulkumar, N. Bhalaji, "Performance analysis of nature inspired load balancing algorithm in cloud environment," Journal of Ambient Intelligence and Humanized Computing, 1–8, 2020, doi: https://doi.org/10.1007/s12652-019-01655-x.

[28] S. Kaur, T. Sharma, "Efficient load balancing using improved central load balancing technique," in 2018 2nd International Conference on Inventive Systems and Control (ICISC), 1–5, IEEE, 2018, doi: 10.1109/ICISC.2018.8398857.

[29] D. Saranya, L. S. Maheswari, "Load balancing algorithms in cloud computing: a review," International Journal of Advanced Research in Computer Science and Software Engineering, **5**(7), 1107–1111, 2015.

[30] A. A. Prakash, V. Arul, A. Jagannathan, "A look at of efficient and more suitable load balancing algorithms in cloud computing," Int. J. Eng. Res. Comput. Sci. Eng., **5**(4), 7, 2018.

[31] P. Prajapati, A. K. Sariya, "A review: Methods of load balancing on cloud computing," Int. J. Res. Anal. Rev., **6**, 2019, doi: 10.6084/m9.doi.one.IJRAR19J2346.

[32] S.-C. Wang, K.-Q. Yan, W.-P. Liao, S.-S. Wang, "Towards a load balancing in a three-level cloud computing network," in 2010 3rd international conference on computer science and information technology, volume 1, 108–113, IEEE, 2010, doi:10.1109/ICCSIT.2010.5563889.

[33] W. Duan, X. Tang, J. Zhou, J. Wang, G. Zhou, "Load balancing opportunistic routing for cognitive radio ad hoc networks," Wireless Communications and Mobile Computing, **2018**, 2018, doi: https://doi.org/10.1155/2018/9412782.

[34] K. Nishant, P. Sharma, V. Krishna, C. Gupta, K. P. Singh, R. Rastogi, et al., "Load balancing of nodes in cloud using ant colony optimization," in 2012 UKSim 14th international conference on computer modelling and simulation, 3–8, IEEE, 2012, doi: 10.1109/UKSim.2012.11.

[35] W. Hashem, H. Nashaat, R. Rizk, "Honey bee based load balancing in cloud computing." KSII Transactions on Internet & Information Systems, **11**(12, doi: 10.3837/tiis.2017.12.001), 2017.

[36] J. Lim, D. Lee, "A Load Balancing Algorithm for Mobile Devices in Edge Cloud Computing Environments," Electronics, **9**(4), 686, 2020, doi:https://doi.org/10.3390/electronics9040686.

[37] P. Kumar, A. Verma, "Scheduling using improved genetic algorithm in cloud computing for independent tasks," in Proceedings of the international conference on advances in computing, communications and informatics, 137–142, 2012, doi: 10.1145/2345396.2345420.

[38] J. AnantKumar, S. PremChand, "Multi-Agent Genetic Algorithm for Efficient Load Balancing in Cloud Computing," in International Journal of Innovative Technology and Exploring Engineering (IJITEE), 45–51, 2020.

[39] R. Panwar, B. Mallick, "Load balancing in cloud computing using dynamic load management algorithm," in 2015 International Conference on Green Computing and Internet of Things (ICGCIoT), 773–778, IEEE, 2015, doi:10.1109/ICGCIoT.2015.7380567.

[40] G. Joshi, S. K Verma, "Load balancing approach in cloud computing using improvised genetic algorithm: a soft computing approach," in IJCA, volume 122, 24–28, 2015, doi: 10.5120/21729-4894.

[41] D. Agarwal, S. Jain, et al., "Efficient optimal algorithm of task scheduling in cloud computing environment," arXiv preprint arXiv:1404.2076, 2014, doi:10.14445/22312803/IJCTT-V9P163.

[42] S. Sidana, N. Tiwari, A. Gupta, I. S. Kushwaha, "NBST algorithm: A load balancing algorithm in cloud computing," in 2016 International Conference on Computing, Communication and Automation (ICCCA), 1178–1181, IEEE, 2016. doi: 10.1109/CCAA.2016.7813914.

[43] M. Yakhchi, S. M. Ghafari, S. Yakhchi, M. Fazeli, A. Patooghi, "Proposing a load balancing method based on Cuckoo Optimization Algorithm for energy management in cloud computing infrastructures," in 2015 6th International Conference on Modeling, Simulation, and Applied Optimization (ICMSAO), 1–5, IEEE, 2015, doi: 10.1109/ICMSAO.2015.7152209.

[44] K. Kumar, T. Ragunathan, D. Vasumathi, P. Prasad, "An Efficient Load Balancing Technique based on Cuckoo Search and Firefly Algorithm in Cloud," Int. J. Intell. Eng. Syst, **13**(3), 422–432, 2020, doi: 10.22266/ijies2020.0630.38.

[45] L. Yan, X. Liu, "The predicted load balancing algorithm based on the dynamic exponential smoothing," Open Physics, **18**(1), 439–447, 2020, doi: 10.1515/phys-2020-0108.

[46] X. Ren, R. Lin, H. Zou, "A dynamic load balancing strategy for cloud computing platform based on exponential smoothing forecast," in 2011 IEEE international conference on cloud computing and intelligence systems, 220–224, IEEE, 2011, doi: 10.1109/CCIS.2011.6045063.