Available Online at www.ijcsmc.com/journal

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 7.056

IJCSMC, Vol. 11, Issue. 11, November 2022, pg.219 – 236

A Review on Load Balancing Algorithms in Cloud Computing

Majd Mutaher Hasan[1]; Nabel Alsohaibi[2]
[1],[2]Faculty of Computer Science, Department of Information Technology, Sana"a University, Sana"a, Yemen
[1] m.izzaldin@su.edu.ye; [2] alsohybe@su.edu.ye

DOI: https://doi.org/10.47760/ijcsmc.2022.v11i11.019

Abstract: Cloud computing is a new standard for large-scale distributed and parallel computing that is still in its infancy. At a predetermined time, provides common resources, information, software packages, and other resources in accordance with client requirements. As cloud computing becomes more popular and more people become interested in assistive computing, better and quicker service is necessary. It is necessary to enhance the management of the good load balancing solutions that are already accessible. Recently, numerous clients from all over the world have been seeking various services at a quick rate, negatively impacting cloud computing performance. The various load balancing techniques used in cloud computing are effective in customizing demand by identifying the appropriate virtual machines.

In this review, a comparative study will be done between many load balancing algorithms in different cloud computing, as many studies have been done on this important topic.

Important research will also be analyzed and defined, and analyzes will be written for each research that talks about dynamic and non-dynamic load balancing techniques and algorithms. Many details will also be mentioned from the researcher's point of view as a summary of each research.

Keywords: Cloud computing, Load balancing, Task scheduling, Virtual machines, Parallel Computing

I. INTRODUCTION

Cloud Computing is a new advanced area with a service-oriented design. The increased use of cloud computing has resulted in an increase in the number of tasks and burden. Due to the unequal distribution of tasks over multiple nodes with varying capacities, this workload creates an imbalanced load. As a result, certain nodes in the cloud are overloaded while others are underloaded, resulting in an uneven load. To make optimal use of cloud resources, it is important to balance this load in order to satisfy the user. Tasks are redistributed from overcrowded nodes to underloaded nodes using the load balancing algorithm. The matrices that impact the load balancing method in the cloud are discussed in this paper. It also includes an examination of several load

© 2022, IJCSMC All Rights Reserved

balancing algorithms as well as related work by other writers. In this work, a method for optimal load balancing in cloud computing is suggested based on research. Cloud computing is a contemporary internet-based service provider that offers clients on-demand access to a shared pool of resources. Hardware or software can be used as resources. Because of its concept of "Everything as a Service," cloud computing has attracted the interest of both academic and industrial groups". Many businesses and people are turning to cloud computing to store huge amounts of data rather than establishing and maintaining their own local data centers, thanks to the growth of cloud computing. In today's society, cloud services are delivered in the same way as water and electricity are. You must pay the same amount as you spent on these services. Cloud users can take use of a variety of public cloud computing services [1]. Users may now concentrate on their primary goal without having to worry about computer resources. The number of cloud computing resources continues to grow with time, resulting in more efficient computer resource use. It is important to enhance efficiency in order to deliver hassle-free services to its clients, which will lead to a rise in throughput. Cloud computing has gotten a lot of attention recently because of the implications for more flexibility and availability at a cheaper cost [2].

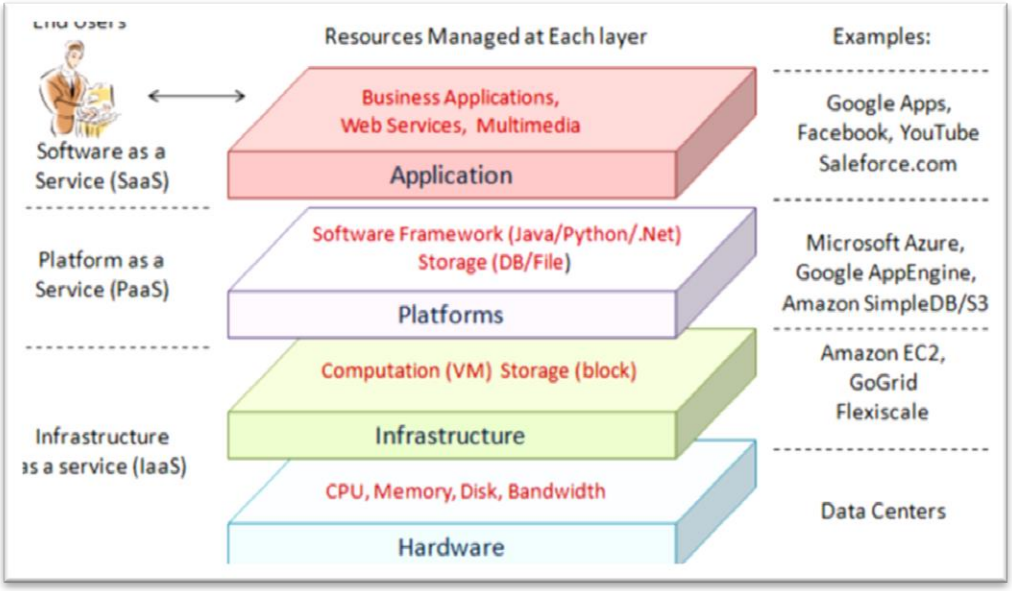
We may highlight improved utilization of dispersed resources and their application to achieve higher productivity and performance and address large-scale computing issues as a primary aim of cloud computing [3].

The architecture of cloud computing (Qi et al. 2010) is divided into four layers as discussed below and shown in Figure 1.1.

The hardware layer is used in data centers to control physical resources. Hardware setup, resource management, fault tolerance, traffic management, power, and cooling are all key challenges at this tier. The virtualization layer, also known as the infrastructure layer, creates a storage and computational resource pool. It partitions physical resources using virtualization technologies, and its fundamental feature is dynamic resource assignment.

The platform layer sits on top of the infrastructure layer and is made up of operating systems (OS) and application frameworks. This layer makes it easier to deploy programs directly into virtual machine containers.

The application layer, which contains the real cloud apps, is the topmost level of the hierarchy. The ability to auto scale to provide availability and greater performance at low operating costs is a critical characteristic of cloud applications.



Source: https://amagsmb.wordpress.com/2014/06/23/cloud-computing-part-3- architecture/

Figure 1.1 Layered Model of Cloud Architecture

© 2022, IJCSMC All Rights Reserved

II. LOAD BALANCING

Load balancing is the act of dynamically and evenly spreading workload across all available nodes in the cloud. By distributing workloads over several nodes, the total system performance increases. When resources are not effectively exploited, they might overheat, resulting in carbon emissions. Carbon emissions can be reduced by effectively using resources [4]. Some measuring metrics that may be used to assess load balancing strategies include throughput, performance, scalability, reaction time, resource usage, and fault tolerance. These characteristics help us to determine if the given load balancing approach or algorithm is capable of balancing the load [5]. Every virtual machine in the cloud system may process the same amount of work thanks to excellent load balancing. As a result, load balancing will be required to increase throughput while reducing response time. It also reduces energy usage, resulting in a cleaner, greener atmosphere...

A) Measurement Parameter for Load Balancing

There are certain measuring criteria that may be used to evaluate load balancing strategies, allowing us to see if the technique is capable of balancing the load [5].

• Throughput: This refers to the quantity of work that may be completed in a particular period of time.

• Response time: This is the time it takes to begin fulfilling the user's demand after registering the request.

• Fault tolerance: This refers to the load balancing algorithm's capacity to function in the event of a system failure.

• Scalability: This refers to the algorithm's capacity to scale up or down in response to changing conditions.

• Performance: This refers to how well the algorithms perform in terms of accuracy and speed.

• Resource utilization: It is used to keep track of how many resources are being consumed.

B) Algorithms for Load Balancing Classification

Load balancing algorithms come in a variety of forms. Load balancing algorithms are often divided into two groups based on the current condition of the system. [6] [7]:

• Static Algorithm: Static Algorithms are useful in environments that are homogenous and steady.

• Dynamic Algorithm: Dynamic Algorithms work well in a diverse setting.

In order to improve the overall performance of a system, load balancing is utilized to transfer a higher processing load among fewer processing nodes. Load balancing is necessary in a cloud computing system to spread the dynamic local workload evenly across all nodes [8]. Load balancing aids in the equitable distribution of computing resources, resulting in high user satisfaction and proper resource use, which reduces resource usage. It aids with failover, scalability, and the avoidance of bottlenecks. Load balancing benefits networks and resources by maximizing throughput while minimizing response time. Furthermore, load balancing splits traffic across all servers, allowing data to be delivered and received without delay. Many techniques are available in a cloud environment to aid in proper traffic loading between all accessible servers. With proper inspections, the majority of them may be connected in a cloud environment. Every processor varies swiftly and effortlessly in the cloud environment, which is a heterogeneous foundation. Heuristic planning computations in online mode are more appropriate and superior for a cloud domain. While constructing a heap adjusting calculation, it is critical to assess acceptable load, evaluate all heaps, the strength of each different framework, execute the purposed framework, cooperation between each of the hubs, and the sort of work to be traded. The most important thing is to choose the hubs, which might include a variety of various ones. The heap of the computer is computed by combining the CPU load and the amount of memory required. Clients may experience a variety of difficulties without adequate load adjusting, such as postponements, timeouts, and delayed framework responses, if effective load adjusting is not implemented.

© 2022, IJCSMC All Rights Reserved

Load balancing is a computer networking technique that distributes workload over many computers or a computer cluster, network links, central processing units, disk drives, or other resources in order to maximize resource utilization, minimize reaction time, and avoid overloading. It is a tool that distributes the dynamic nearby workload evenly across each of the cloud's hubs in order to improve client satisfaction and asset utilization, therefore improving the framework's overall general performance and asset utility. By eliminating bottlenecks and fail-over, it also ensures that each processing asset is distributed productively and fairly. Static and dynamic load balancing are the two types of load balance [9].

III. METHODOLOGY

The methodology that used in this thesis as the following:

1- Literature Survey.

2- Studies on different methods and algorithms. Theoretical analysis of the **existing Algorithms.**

CLOUD INVESTIGATION OF LOAD BALANCING ALGORITHMS:

The performance of three algorithms with distinct Service Broker rules, namely Round Robin, Equally Spread Current Execution (ESCE), and Throttled load balancing, was compared by Thakur and Kumar (2015). The simulations were conducted using the CloudSim tool, which found that controlled load balancing with the Optimistic Service Broker policy was the most effective. Due to lower migration overheads, the Round Robin algorithm under the Closest Service Broker policy proved particularly cost-effective.

Ariharan & Manakattu (2015) proposed the Neighbour Aware Random Sampling (NARS) algorithm, which selects the least loaded node from the neighbour list for the random walk. To improve the node selection process, neighbour awareness and prediction algorithms are applied. This approach results in a higher likelihood of selecting a lightly loaded node and a shorter job wait time.

Pan et al. (2015) devised an optimal load balancing technique that improves system production and efficiently allocates workloads to VMs. It was discovered that the key to overcoming load balancing difficulties is a perfect scheduling algorithm that not only balances the load but also meets the needs of the users. The aforementioned approach has demonstrated that the time it takes for all tasks in a system to be completed is the same.

Kashyap and Viradiya (2014) provided a comprehensive overview of cloud load balancing algorithms, including their benefits, drawbacks, and performance indicators. Different algorithms were compared in this article based on various load balancing factors.

Chen et al. (2014) introduced Resource Intensity Aware Load Balancing (RIAL), a resource weight based approach. The resource weight was calculated using this method, which was based on the intensity of resources when VMs were migrated from strongly loaded Physical Machines (PMs) to lighter loaded PMs. The approach resulted in a lower-cost, faster convergence to load balancing, as well as a lower likelihood of future load imbalance.

Xu et al. (2013) proposed a public cloud load balancing strategy based on the cloud partitioning concept with a switching mechanism.

© 2022, IJCSMC All Rights Reserved

Kaur et al. (2013) introduced a 2-Phase Load Balancing method that was a combination of Opportunistic Load Balancing (OLB) and Load Balance Min-Min (LBMM) scheduling algorithms that maintains a balanced load while maximising executing efficiency. The OLB algorithm maintains all nodes in a working state in order to fulfil the goal, while the LBMM method is used to lower the execution time of each task and the total completion time. The resource consumption was improved thanks to this technique.

Mulay and Jain (2013) prioritised requests and suggested an Enhanced Equally Distributed Load Balancing method that was run by a single central server node. The load was distributed not just in a balanced manner, but also in a systematic manner by validating the load.
Grover & Katiyar (2013) employed the Agent Based Dynamic Load Balancing (ABDLB) approach, in which a mobile agent, an independent software programme, executes on behalf of a network administrator. When compared to standard load balancing techniques, the suggested agent-based method provides lower communication costs, higher throughput, and faster response times.

When a request arrives, the Active Monitoring Load Balancer (AMLB) retains information about each VM and identifies the least loaded VM, according to James & Verma (2012). A Weighted Active Monitoring Load Balancer was proposed as an upgraded AMLB that distributes weights to each virtual machine. It keeps track of VMs, their weights, and the number of requests currently assigned to each of them in an index table. The algorithm improved response time and accuracy.

Kaur proposed the Equally Spread Active Execution (ESAE) algorithm (2012). The first stage in this approach is to use Cloud Manager to determine the availability of VMs and validate their current load. If the load is less than the threshold, the job is assigned to the VM right away. The computer system was load balanced since the jobs are evenly distributed, and no VMs are underutilised. The ESAE algorithm shortens response and processing times. It also reduced the cost of virtual machines and data transfer.

To reduce cloud power consumption, Galloway et al. (2011) developed a Power Aware Load Balancing (PALB) algorithm. Its purpose was to keep computational nodes available while reducing total power usage. The technique was implemented in a Cluster Controller, which keeps track of each current node's use and decides where new VMs should be created. The benefit of this technique was that it turned off idle physical equipment, but the problem was that it made it impossible to grow up massive cloud data centres.

Mehta et al. (2011) proposed a Content Aware Decentralized Load Balancing Algorithm based on Workload and Client Aware Policy (WCAP). The unique and special properties of the requests were specified using a Unique and Special Property (USP), which aids the scheduler in determining the right node for processing the requests. This technique was implemented in a decentralised manner with minimal overhead, and it increases utilisation by minimising computing node idle time.

Xu et al. (2011) introduced a Token Based Load Balancing mechanism in which overcrowded DCs must request computational resources from other DCs, and jobs are transferred to the new DC with more resources. When a DC generates a token, it is claiming the request for a job to be transferred from an overloaded node. When a DC receives a token from an overloaded DC, it must decide whether to accept or reject it. If it accepts, it assigns the work to the desired resource. Otherwise, the token is rejected. Once the token was approved, the job was moved to the DC.

© 2022, IJCSMC All Rights Reserved

Padhy and Rao (2011) devised a load balancing technique based on the Divisible Load Scheduling theorem to maximise or minimise different performance characteristics such as throughput and latency of cloud of various sizes, i.e. virtual topology, depending on the application requirement.

Banerjee (2011) incorporated an Elastic Load Balancing method into StarCluster to overcome the flaw. To boost work throughput, the Elastic Load Balancer (ELB) adds instances to the cluster and terminates instances to save money and energy. The ELB will analyse its workload, poll the cluster on a regular basis, and then decide whether the cluster should be updated. As a result, the StarCluster with ELB has increased job throughput during peak periods while saving money during off-peak periods.

According to Chaczko et al. (2011), load balancing approaches increase cloud resource use and availability. A Message Oriented Model (MOM) was employed in this study, in which messages were used to communicate across the network. Online instant chat programmes mostly used the EXtensible Message Passing Protocol (XMPP). XMPP is a communications protocol that allows for resource monitoring and availability in the cloud.

Amar et al. (2011) researched an adaptive resource management for Web applications using a heterogeneous test bed compute cloud. The suggested system was built on top of the EUCALYTPUS framework in order to dynamically distribute resources to the Web application and measure its response time. A novel type of Service Level Agreement (SLA) for compute clouds has been uncovered in the research, in which the customer wants a particular response time for the Web application.

Alakeel (2010) investigated the creation of a dynamic load balancing algorithm that addresses a variety of challenges, including load estimation, performance index, system stability, data sharing among nodes, resource estimation, job and remote node selection, and more. The aforementioned concerns, which should be considered in the construction or study of the dynamic load balancing algorithm, have been given and examined in this work.

Wee and Liu (2010) looked into the possibility of setting up a large-scale web server farm in the cloud. A client-side load balancing architecture was presented, capable of scaling and handling faults in milliseconds. The architecture's high throughput was demonstrated experimentally in this study.

A REVIEW OF CLOUD TASK SCHEDULING ALGORITHMS

Ali and Alam (2016) published a study of various Cloud-based Task Scheduling algorithms that provide scalability, on-demand resources, reduced energy usage, and improved cloud system performance. Various scheduling characteristics such as execution time, throughput, makespan, resource utilisation, response time, and cost have been examined and compared in this article.

Madni et al. (2016) looked into several resource allocation techniques and schemes. In DCs or VMs, load balancing is accomplished by allocating resources and sharing loads in a methodical manner to achieve high performance and resource utilisation. The VM allocation algorithm assigns VMs to DC hosts, whereas the Cloudlet allocation algorithm assigns cloudlets to VMs using load balancing methods, ensuring that each cloudlet completes its task quickly and efficiently.

© 2022, IJCSMC All Rights Reserved

Panda and Jana (2015) proposed a multi-cloud task scheduling algorithm that takes into account essential task scheduling characteristics such as makespan, average resource consumption, and cost of service. Min-Min scheduling reduces service costs and Profit Based Task Scheduling (PBTS) minimises makespan.

Kaur and Kinger (2014) evaluated various Job Scheduling algorithms such as FCFS, SJF, Priority, and Round Robin to explore basic scheduling challenges in the cloud context. Each algorithm was found to have some flaws, necessitating the creation of a new scheme that met all of the criteria while also providing superior performance.

Mathew et al. (2014) conducted a comprehensive analysis of several work scheduling strategies and their parameters. Heuristic, energy-efficient, and hybrid approaches are also included in the research. Using multiple metrics, a better scheduling algorithm can be constructed from existing approaches, resulting in improved performance.

Using the CloudSim toolkit, Priyadarsini and Arockiam (2014) studied the performance of Min-Min and Min-Max scheduling algorithms. Sixty tasks were allocated to the cloud with four resources, and the makespan was calculated. It was discovered that max-min scheduling achieves a better makespan for tiny distributed systems.

Annette et al. (2013) discovered that task dependency has a significant impact on the design of a scheduling method. The paper provides an overview of several scheduling strategies, with a focus on task dependency.

Marphatia et al. (2013) proposed an improved version of the FCFS scheduling method to meet the primary issues of cloud task scheduling. The proposed model was built and evaluated on a simulation toolkit with the goal of creating a module that represents the standard FCFS method and comparing it to the optimised approach for cloud resource provisioning. From both the server and client perspectives, the experiment revealed optimised memory utilisation and improved system performance.

Mohana Priya & Subramani (2013) proposed a Resource Aware Scheduling Algorithm (RASA) that was a hybrid of Min-Min and Max-Min algorithms. If the number of available resources was odd, the Min-Min approach was used to allocate jobs; otherwise, the Max-Min technique was employed. The Min-Min and Max-Min strategies result in the execution of small and big tasks on various resources in a sequential order. The suggested approach provides optimal resource allocation for jobs, as well as reduced execution time, resulting in improved resource use and performance.

Elzeki et al. (2012) created an Improved Max-Min method with a one-of-a-kind change to the Max-Min algorithm. The main concept was to distribute jobs with the longest execution times to the resource with the shortest completion times. It starts with an unplanned set of tasks and then calculates the predicted execution matrix and completion time for each task based on the available resources.

Eludiora et al. (2010) used Java Development Environment (JADE) to create a strategy called "Cooperative Adaptive Symmetrical Initiated Dynamic / Diffusion (CASID)" to handle uncontrolled job/task migration among servers. Because it is critical to manage bandwidth consumption while load/tasks are being transmitted between servers, bandwidth optimization was explored in this research. The purpose of this project was to

© 2022, IJCSMC All Rights Reserved

standardise job migration in order to reduce bandwidth usage. In terms of response time and system throughput, the CASID policy outperformed.

A three-level hierarchical architecture was used by Wang et al. (2010) to propose a Load Balance Min-Min (LBMM) scheduling algorithm. The request manager is the initial level, and it is responsible for assigning the assignment to an appropriate service manager. The service manager is the second level, which is used to break the task into logically distinct subtasks. The service nodes, which are utilised to execute subtasks, are the third level. The proposed LBMM method addressed load imbalance and decreased each node's execution time.

Analysis on Meta-heuristic Scheduling Algorithms

Remesh Babu and Samuel (2016) suggested an Enhanced Bee Colony method to schedule and balance the load across computational nodes in a dynamic cloud environment. The Swarm Intelligence method was utilised to remove work from overloaded VMs and submit them to underloaded VMs in this paper, and the honey bee foraging behaviour was employed to balance the load across VMs. This strategy reduces response time and completion time while minimising the makespan and number of VM migrations.

MSOACS is a study by Durgadevi and Srinivasan (2015) that combines a meta-heuristic task scheduling algorithm based on Swarm Optimization (SO) and Cuckoo Search (CS). MSOA, which deals with the problem of VM placement and task scheduling in the cloud, was the subject of this paper. The suggested technique was tested using the CloudSim tool, which resulted in a shorter maketime and a higher resource usage ratio.

Joshi and Verma (2015) suggested a soft computing approach called the Improvised Genetic Algorithm (IGA). It allocates incoming work to servers or virtual machines using a natural selection technique. The cost value is treated as a fitness function of an individual node while doing load balancing. When compared to the Genetic Method (GA) and the Round Robin algorithm, it was simulated using the MATLAB toolbox and CloudAnalyst, which yields a faster reaction time.

Cho et al. (2015) integrate Ant Colony Optimization (ACO) with Particle Swarm Optimization (PSO) to tackle the VM scheduling problem, resulting in ACO with Particle Swarm Optimization (ACOPS). To adjust changing circumstances, ACOPS uses historical data to forecast the burden of future requests. The scheduling procedure's computation time has been lowered using this way. The suggested method outperforms existing algorithms in terms of maintaining a balanced load in a dynamic environment.

Dam et al. (2015) proposed a novel load balancing technique in which the overloaded node searches for an underloaded node to balance the load. When the proposed method's results were compared to the standard strategy, local search algorithm, and soft computing approaches, it outperformed them all.

Florence and Shanthi (2014) suggested a method that maximised resource use and provided a well-balanced load. This study used the Firefly Technique (FA), a modern optimization algorithm, to perform load balancing operations based on the load index. The proposed approach efficiently optimises schedules with a balanced load, according to the paper.

Pacini et al. (2013) described and tested a PSO-based cloud scheduler. As their work centred on the cloud IaaS model, the publication addressed a study on how private clouds perform scientific studies.

© 2022, IJCSMC All Rights Reserved

Yang (2012), Yang (2010a), Yang (2010b), and Yang (2009) aimed to give a full description of a new Firefly Algorithm (FA) for multimodal optimization. When compared to other meta-heuristic algorithms, such as PSO, the FA was found to be potentially more powerful in tackling NP-Hard problems.

FINDINGS OF LOAD BALANCING AND SCHEDULING TECHNIQUES IN CLOUD COMPUTING:

The primary goal of this in-depth analysis is to provide a systematic review of existing cloud load balancing strategies, with the goal of contributing an efficient load balancing algorithm to balance and schedule the load to the available resource. Table 2.2 shows the results of several load balancing approaches in the cloud environment. Table 2.3 summarises the many measures used to evaluate load balancing techniques.

Review on Load Balancing and Scheduling Algorithms

S. No.	Algorithm / Technique	Description	Findings
1	Enhanced Bee Colony algorithm	Honey bee foraging behaviour used to balance the load across VMs	• Minimized makespan and number of VM migrations • Response time and completion time are minimized
2	Comparison of Select LB algorithms	The performance of the three algorithms with various Service Broker policies is compared	• Cost-effective
3	NARS algorithm	Neighbour awareness and prediction mechanisms are used	• Increased probability of choosing lightly loaded node • Reduced Job waiting time
4	Artificial Bee Colony based LB method	To solve the load balancing problems and to meet the users‟ needs	• Tasks scheduled to VMs more efficiently
5	Improvised Genetic Algorithm	The soft computing approach uses Natural Selection strategy	• Gives better response time when compared to GA and RR algorithm
6	ACO with Particle Swarm (ACOPS)	The historical information is used to predict the workload	Solves the VM Scheduling problem • Reduces the computing time of scheduling
7	Meta-heuristic Task Scheduling Algorithm	Combines Swarm Optimization and Cuckoo Search algorithms	• Reduces makespan • Improves utilization ratio

© 2022, IJCSMC All Rights Reserved

8	Hybrid LB strategy	Underloaded nodes are searched for balancing load with overloaded Node	• Compared and outperformed the existing strategies like FCFS, SHC, GA and ACO
9	Dynamic Scheduling based on PSO	Focus on IaaS model which launch VMs to hosts available and allocate VMs to Physical resources	• Cloud scheduler succeeds in balancing based on Random assignment and GA

S. No.	**Algorithm / Technique**	**Description**	**Findings**
10	LB model using FA	Performs load balancing operation based on load index	• Improved resource utilization • Good balanced load
11	LB Model based on Cloud Partitioning	Balances the load by partitioning the cloud using Main Controller and Load Balancer	• Managing the cloud environment is easy • Cloud Division rules are not specified
12	2-Phase Load Balancing Algorithm	OLB is used to keep each node busy whereas LBMM is used to achieve a minimum execution time of each Job	• Efficient resources utilization
13	Enhanced Equally Distributed LB algorithm	The algorithm is executed by the Central Server node which allocates counter variable to each DC	• Systematically allocates the load and balance the load • Overhead on the Central server • Single point of failure
14	ABDLB approach	Mobile agent plays a vital role	• Reduced communication cost • Improved load balancing, throughput and response time
15	Optimization of FCFS based Resource Provisioning algorithm	Dynamically allocates resources depending on the specification of each user request	• Enables optimized memory usage • Enhances system performance

© 2022, IJCSMC All Rights Reserved

16	RASA algorithm	Combine Min-Min and Max-Min strategy which results in the consecutive execution of tasks on different resources	• Efficient resource allocation for tasks • Minimized execution time • Better performance and resource utilization
17	Weighted AMLB	Assigns weight to each VMs and maintains index table of VM and its associated weight	• Achieve a better response time and processing time.

S. No.	Algorithm / Technique	Description	Findings
18	Improved Max- Min algorithm	Assign task with maximum execution time to resource that produces minimum completion Time	• Larger tasks are executed quickly • Small jobs must wait for a long time
19	ESAE algorithm	The Cloud Manager estimates the size of the job, verifies the availability and capacity of VMs	• Better response time and processing time • No VM is underutilized • Cloud Manager gets a bottleneck problem & single point of failure
20	Decentralized Content Aware Load Balancing Algorithm	USP of nodes is used and the content information is used to narrow down the search	• Improves overall performance • Reduces idle time of nodes
21	Join-Idle Queue Algorithm	Idle Processors are assigned to the dispatchers and then jobs are assigned so that the average queue length will be reduced	• Reduced system load • Less communication overhead
22	Honey Bee Foraging Behaviour	Global load balancing is achieved through local server actions	• Improved scalability
23	MOM and XMPP	MOM improves load balancing through messaging and XMPP allow resources monitoring and provide resource availability	• Better improvements in resource utilization and availability

© 2022, IJCSMC All Rights Reserved

24	SLA Driven LB	Acts as a heterogeneous test bed compute cloud and investigates adaptive management of resources of Web applications	• Monitors the response time of computed resources • Dynamically allocates the resources
25	CASID policy	Addresses unregulated jobs/tasks migration and Bandwidth optimization is considered	• Better response time and system throughput

S. No.	Algorithm / Technique	Description	Findings
26	Client-Side Load Balancer	The browser decides to allocate jobs to servers based on cache information	• Client based Load balancing & no server overhead • High throughput • Cache information is not updated dynamically
27	PALB Algorithm	Implemented in the Cluster Controller and uses Job Scheduler to simulate user requests for VM instances	• Idle state Physical Machines are moved to power off state to conserve energy • It does not scale up for large data centers
28	Task Scheduling based on LB	Map the tasks to VMs and then to host resource	• Improves the response time and resource utilization
29	Central Load Balancing Policy for Virtual Machines (CLBVM)	The global state information is used to make load balancing decisions	• Balance the load evenly to improve system performance • Does not consider fault tolerance
30	CloudSim	The toolkit enables modeling and simulation of Cloud systems	• Tests newly developed methods and policies
31	CloudAnalyst	Simulates large-scale Cloud applications	• Used to identify new features and issues, model them • Develop and evaluate new algorithms for resource management

© 2022, IJCSMC All Rights Reserved

Comparison of Existing Load Balancing Techniques based on various parameters

S.No.	Author / Year	Algorithm	Waiting Time	Execution Time	Response Time	Completion Time	Makespan	Load Balancing	Bandwidth	Throughput	Resource Utilization	Power Consumption	Cost	Memory Usage	Performance	QoS
1	Remesh Babu & Samuel 2016	Enhanced Bee Colony			*	*	*									*
2	Ali & Alam 2016	Comparison of Task Scheduling			*	*	*			*	*		*			
3	Ariharan & Sheeja 2015	Neighbour Aware Random Sampling (NARS)	*					*								
4	Pan et al. 2015	Optimal Load Balancing				*		*								
5	Durgadevi & Srinivasan 2015	Metaheuristic Swarm Optimization Algorithm with Cuckoo Search (MSOACS)					*				*					
6	Joshi & Verma 2015	Improvised Genetic Algorithm (IGA)			*			*								
7	Panda & Jana 2015	Cloud Min- Min Scheduling & Profit Based Task Scheduling (PBTS)					*						*			

© 2022, IJCSMC All Rights Reserved

S.No.	Author / Year	Algorithm	Waiting Time	Execution Time	Response Time	Completion Time	Makespan	Load Balancing	Bandwidth	Throughput	Resource Utilization	Power Consumption	Cost	Memory Usage	Performance	QoS
8	Cho et al. 2015	Ant Colony Optimization with Particle Swarm Optimization (ACOPS)	*					*								
9	Dam et al. 2015	Novel Load Balancing strategy						*								
10	Priyadarsi ni & Arockiam 2014	Min-Min and Min-Max Scheduling					*									
11	Florence & Shanthi 2014	Firefly Algorithm (FA)						*								
12	Chen et al. 2014	Resource Intensity Aware Load balancing (RIAL)						*					*			
13	Xu et al. 2013	LB model for the Public Cloud based on the Cloud partitioning						*								
14	Kaur et al. 2013	2-Phase Load Balancing		*				*		*	*					
15	Mulay & Jain 2013	Enhanced Equally Distributed Load Balancing						*								
16	Grover & Katiyar 2013	Agent Based Dynamic Load Balancing (ABDLB)			*			*		*						

© 2022, IJCSMC All Rights Reserved

S.No.	Author / Year	Algorithm	Waiting Time	Execution Time	Response Time	Completion Time	Makespan	Load Balancing	Bandwidth	Throughput	Resource Utilization	Power Consumption	Cost	Memory Usage	Performance	QoS
17	Marphatia et al. 2013	Optimized FCFS Scheduling												*	*	
18	Mohana Priya & Subramani 2013	Resource Aware Scheduling (RASA)		*							*				*	
19	James & Verma 2012	Weighted Active Monitoring Load Balancer (AMLB)		*	*											
20	Elzeki et al. 2012	Improved Max-Min Algorithm		*		*										
21	Kaur 2012	Equally Spread Active Execution (ESAE)		*	*								*			
22	Mehta et al. 2011	Decentralized Content aware Load Balancing	*								*				*	
23	Xu et al. 2011	Token Based Load Balancing		*												
24	Padhy & Rao 2011	Effective Load Balancing Algorithm using Divisible Load Scheduling theorem													*	
25	Banerjee 2011	Elastic Load Balancing mechanism		*						*			*			

© 2022, IJCSMC All Rights Reserved

S.No.	Author / Year	Algorithm	Waiting Time	Execution Time	Response Time	Completion Time	Makespan	Load Balancing	Bandwidth	Throughput	Resource Utilization	Power Consumption	Cost	Memory Usage	Performance	QoS
26	Chaczko et al. 2011	Message Oriented Model (MOM)		*							*					
27	Amar et al. 2011	Heterogeneous Test Bed			*											
28	Galloway et al. 2011	Power Aware Load Balancing (PALB)		*							*	*				
29	Lu et al. 2011	Join-Idle-Queue			*										*	
30	Eludiora et al. 2010	Cooperative Adaptive Symmetrical Initiated Dynamic / Diffusion (CASID)			*				*	*						
31	Wee & Liu 2010	Client Side Load Balancing						*		*						*
32	Wang et al. 2010	Load Balance Min-Min (LBMM) Scheduling		*				*								
33	Fang et al. 2010	Task Scheduling based on LB			*						*				*	
34	Randles et al. 2010	Honeybee Foraging Behaviour								*					*	
35	Hu et al. 2010	Scheduling strategy on LB of VM resources									*					

© 2022, IJCSMC All Rights Reserved

IV. Conclusions

This review summarized many previous research related to load balancing in cloud computing, as well as the names and types of algorithms that worked and were employed in load balancing shorter.

REFERENCES

[1]. Mollah, M. B., Islam, K. R., & Islam, S. S. (2012, April). Next generation of computing through cloud computing technology. In 2012 25th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE) (pp. 1-6). IEEE.

[2]. Botta, A., De Donato, W., Persico, V., & Pescapé, A. (2016). Integration of cloud computing and internet of things: a survey. Future generation computer systems, 56, 684-700.

[3]. Agelastos, A., Allan, B., Brandt, J., Cassella, P., Enos, J., Fullop, J., ... & Tucker, T. (2014, November). The lightweight distributed metric service: a scalable infrastructure for continuous monitoring of large scale computing systems and applications. In SC'14: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (pp. 154-165). IEEE.

[4]. Pahl, C., Jamshidi, P., & Weyns, D. (2017). Cloud architecture continuity: Change models and change rules for sustainable cloud software architectures. Journal of Software: Evolution and Process, 29(2), e1849.

[5]. Pahl, C., Jamshidi, P., & Weyns, D. (2017). Cloud architecture continuity: Change models and change rules for sustainable cloud software architectures. Journal of Software: Evolution and Process, 29(2), e1849.

[6]. Diaby, T., & Rad, B. B. (2017). Cloud computing: a review of the concepts and deployment models. International Journal of Information Technology and Computer Science, 9(6), 50-58.

[7]. Tadapaneni, N. R. (2017). Different Types of Cloud Service Models.

[8]. Manohar, N. (2013). A survey of virtualization techniques in cloud computing. In Proceedings of international conference on vlsi, communication, advanced devices, signals & systems and networking (vcasan-2013) (pp. 461-470). Springer, India.

[9]. Shaw, S. B., & Singh, A. K. (2014, September). A survey on scheduling and load balancing techniques in cloud computing environment. In 2014 international conference on computer and communication technology (ICCCT) (pp. 87-95). IEEE.

[10]. Schoonderwoerd, R., Holland, O. E., Bruten, J. L., & Rothkrantz, L. J. (1997). Ant-based load balancing in telecommunications networks. Adaptive behavior, 5(2), 169-207.

[11]. There are certain measuring criteria that may be used to evaluate load balancing strategies, allowing us to see if the technique is capable of balancing the load

[12]. Jyoti, A., Shrimali, M., Tiwari, S., & Singh, H. P. (2020). Cloud computing using load balancing and service broker policy for IT service: a taxonomy and survey. Journal of Ambient Intelligence and Humanized Computing, 11(11), 4785-4814.

[13]. Razzaghzadeh, S., Navin, A. H., Rahmani, A. M., & Hosseinzadeh, M. (2017). Probabilistic modeling to achieve load balancing in expert clouds. Ad Hoc Networks, 59, 12-23.

[14]. Sidhu, A. K., & Kinger, S. (2013). Analysis of load balancing techniques in cloud computing. International Journal of computers & technology, 4(2), 737-741.

[15]. Bezawada, A., Marella, S. T., & Gunasekhar, T. (2018). A systematic analysis of load balancing in cloud computing. International Journal of Simulation--Systems, Science & Technology, 19(6).

[16]. Milani, A. S., & Navimipour, N. J. (2016). Load balancing mechanisms and techniques in the cloud environments: Systematic literature review and future trends. Journal of Network and Computer Applications, 71, 86-98.

[17]. Afzal, S., & Kavitha, G. (2019). Load balancing in cloud computing–A hierarchical taxonomical classification. Journal of Cloud Computing, 8(1), 1-24.

[18]. Sufiev, H., & Haddad, Y. (2016, November). A dynamic load balancing architecture for SDN. In 2016 IEEE International Conference on the Science of Electrical Engineering (ICSEE) (pp. 1-3). IEEE.

[19]. Taghian, S., & Nadimi-Shahraki, M. H. (2019). Binary sine cosine algorithms for feature selection from medical data. arXiv preprint arXiv:1911.07805.

[20]. Yang, X. S. (2009). Harmony search as a metaheuristic algorithm. In Music-inspired harmony search algorithm (pp. 1-14). Springer, Berlin, Heidelberg.

[21]. Nguyen, T. T., Pham, T. D., Kien, L. C., & Van Dai, L. (2020). Improved coyote optimization algorithm for optimally installing solar photovoltaic distribution generation units in radial distribution power systems. Complexity, 2020.

[22]. Shakeabubakor, A. A., Sundararajan, E., & Hamdan, A. R. (2015). Cloud computing services and applications to improve productivity of university researchers. International Journal of Information and Electronics Engineering, 5(2), 153.

© 2022, IJCSMC All Rights Reserved

[23]. Aruna, M., Bhanu, D., & Karthik, S. (2019). An improved load balanced metaheuristic scheduling in cloud. Cluster Computing, 22(5), 10873-10881.

[24]. Ariharan, V., & Manakattu, S. S. (2015, March). Neighbour Aware Random Sampling (NARS) algorithm for load balancing in Cloud computing. In 2015 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT) (pp. 1-5). IEEE.

[25]. He, K., Rozner, E., Agarwal, K., Felter, W., Carter, J., & Akella, A. (2015). Presto: Edge-based load balancing for fast datacenter networks. ACM SIGCOMM Computer Communication Review, 45(4), 465-478.

[26]. Abazari, F., Analoui, M., Takabi, H., & Fu, S. (2019). MOWS: multi-objective workflow scheduling in cloud computing based on heuristic algorithm. Simulation Modelling Practice and Theory, 93, 119-132.

[27]. Chen, L., Shen, H., & Sapra, K. (2014, November). Distributed autonomous virtual resource management in datacenters using finite-markov decision process. In Proceedings of the ACM Symposium on Cloud Computing (pp. 1-13).

[28]. Madni, S. H. H., Latiff, M. S. A., Ali, J., & Abdulhamid, S. I. M. (2019). Multi-objective-oriented cuckoo search optimization-based resource scheduling algorithm for clouds. Arabian Journal for Science and Engineering, 44(4), 3585-3602.

© 2022, IJCSMC All Rights Reserved