2022 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COM-IT-CON) | 978-1-6654-9602-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/COM-IT-CON54601.2022.9850842

Comparative Analysis of Various Load Balancing Techniques in Cloud Environment

Shaik Muhammad Irfan, Hemanth Rathore, Harsh Bisen, Dheeraj Kumar
Suresh Kumar, Kamlesh Sharma
Manav Rachna International Institute of Research and Studies Faridabad, Haryana
irfanskrm@gmail.com, rathorehemant@gmail.com, bisenharsh786@gmail.com, dheeraj322000@gmail.com,
suresh.fet@mriu.edu.in, Kamlesh.fet@mriu.edu.in

***Abstract*— Today, cloud computing has emerged as new technology and a business model. Cloud computing is spreading everywhere on the planet because of its simplicity and easy-to-use model. Growing cloud computing services offer great opportunities for sponsors to induce the most effective and the best prices, which poses new challenges in choosing the simplest service for an outsized group. Cloud computing uses a spread of computer resources to facilitate large-scale operations. Therefore, selecting the correct node to try to do the duty can improve the performance of an outsized cloud computing site. It takes time for consumers to assemble the required information and analyze all service providers to create a call.. Load balancing is the technique used to distribute the loads among various systems to optimize performance. Load balancing ensures that nodes are neither overloaded nor underloaded. Cloud Analyst could be a tool that helps developers to simulate large-scale Cloud applications to understand the functionality of such programs under various deployment settings. The cloud analyst simulator is used to run the experiments.**

Keywords—Cloud computing; Load balancing; Round Robin; Equally Spread Current Execution; Throttled; Response time minimization; Cloud Analyst.

I. INTRODUCTION

Cloud computing is the most helpful technology used in computer science. A report said that the cloud made a big difference in the users' lifestyles by providing the best services. Cloud provides services by enabling universal, convenient, on-demand network access to a shared pool of resources, e.g. servers, storage, applications, and services that can be quickly equipped and released with minimum efforts of management or service providers. Cloud computing is a virtual framework that gives shared information and communication services via the cloud.

Cloud computing provides access to data without requiring ownership of the infrastructure. Technology is advancing and powerful day by day, and it works on the principle of centralization of resources with an on-demand and pay-as-you-go policy [4].

The five most essential applications of clouds include On-demand self-service, global network access, distributed pooling of resources, measured services, and scalability. Cloud services are classified into three categories based on the cloud deployment environment: public cloud, hybrid cloud, and private cloud.

Public clouds are available to the clients from a third-party service provider via the internet, e.g., Google, Gmail, Amazon, etc.

Hybrid clouds mean combining two clouds (public, internal, private, or external) with a virtualized server. A private cloud is an environment where a single customer has access to all hardware and software resources. Cloud computing gives the best services like (SaaS) software as a service. Users can use any software application from different servers via the internet, accessing web browsers without purchasing, e.g., Gmail, Salesforce.com, etc.
Platform as a service (PaaS) in which the development of the framework is offered to the developers for the fast deployment of their code. For e.g. PaaS includes google app engine , Heroku, cloud foundry,etc.

(LaaS) i.e least infrastructure as a service here, the shared infrastructure such as storage, servers, and network are delivered over the internet. e.g. amazon web services, etc.

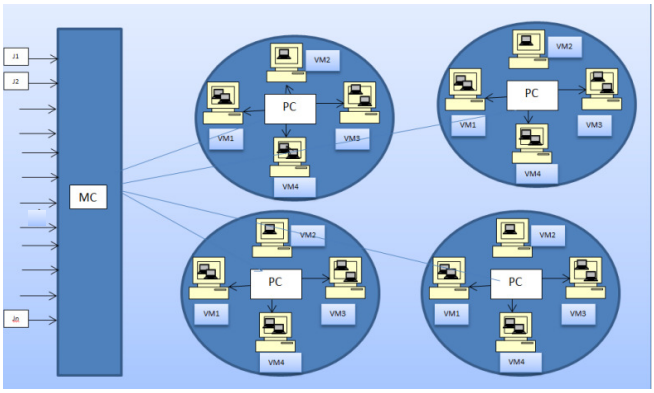

Fig 1. Load balancing in cloud computing

This research paper consider a cloud computing environment as shown in figure 1, where having Pn number of partitions of datacentres denoted as P={P1, P2, P3……Pn}, j=jobs, which is monitored by the main controller MC. Each partition of datacentre has i number of virtual machines (VM) denoted as VM= {VM1,VM2,VM3……VMi} which is monitored by partition control PC. The Main controller (MC) receives the k number of jobs and the main controller assign the jobs to the partitions of data centres which are either in idle or normal state. The partition controller send the load status to data centre LS={i, n, o} where i is idle, n is normal and o is overloaded. Therefore main controller will assign the incoming jobs to the partition controller, and then the partition controller will run the algorithms and allocate the jobs to the virtual machines

.

II. LITERATURE REVIEW

Sambit Kumar Mishra, Bibhudatta Sahoo, Priti Paramita Parida [2] considered the basics of load balancing and some commonly used techniques for load balancing in homogenous and non-homogenous cloud domains. They have analyzed the performance of algorithms using the CloudSim simulator.

Bhargavi K, Sathish Babu B, and Jeremy Pit [13] have analyzed the performance of Swarm Artificial Intelligence-based load balancing techniques in cloud domainslike Whale, dragonfly, spider, and raven. The comparison criteria are total execution time, response time, resource utilization rate, and throughput. They have also used CloudSim simulator under three distinct situations: data center partitions variation, job variation, and evaluation of different jobs. They conclude that the performance of the raven roosting algorithm and dragonfly algorithm is better as they circulate the load among the VMs with less execution time, efficient resource utilization, less response time, and at a high rate of task completion.

Sawsan Alshattnawi and Mohammad AL-Marie [15], analyzed the Spider Monkey Optimization Inspired Load Balancing (SMO-LB). This technique targets to balance the loads among the virtual machines (VMs) by lowering the makespan and response time. The performance of this technique is determined using the cloudSim simulator under distinct situations in comparison with the Round Robin method and Throttled method. The experiment shows that this technique takes 10.7 seconds of average response time while the Round Robin and Throttled methods take 24.6 and 30.8 seconds respectively, which is comparably higher than the spider technique. Also, the make span observed in (SMO-LB) is 21.5 seconds which is lower than Round Robin and Throttled methods which take 35.5 and 53.0 seconds respectively.

Azizkhan F Pathan [16] analyzed the load balancing algorithms using cloud analyst. The traffic is distributed evenly among servers using the static method. This algorithm necessicates prior knowledge of system resources, ensuring that the decision to shift the load is independent of the systems current state. In a system with little volatility in demand, a static algorithm is appropriate. A dynamic algorithm searches for and prefer the lightest server in the entire network.

This necessaciates a realtime network connection, which can significantly increase system traffic. The current state of the system is used to make judgements on how to manage the load.

Cloud computing employs a virtualization approach that divides a single system into several virtual ones. Load balancing determines which clients will use the virtual machine and which will be placed on hold. Virtualization technology allows for dynamic load balancing of the entire system, allowing for the remapping of virtual machines (VMs) and physical resources in response to changes in load. Virtualization technology is being widely used in Cloud computing as a result of these benefits. A virtual machine (VM) is a software implementation of a computing environment that can be used to install and run an operating system (OS) or programme. The Virtual Machine alters the physical computing environment by requesting CPU, memory, hard disc, and network resources. Calls for CPU, memory, hard drive, network, and other hardware resources are controlled by a virtualization layer, which translates these requests to the underlying physical hardware Virtual machines (VMs) are constructed using a virtualization layer that runs on top of a client or server operating system, such as a hypervisor or a virtualization platform. The host OS is the name given to this operating system. The virtualization layer can be used to create numerous separate, separated virtual machine environments in which multiple requests or processes can run simultaneously on multiple machines.

III. DYNAMIC LOAD BALANCING ALGORITHMS

A. Round Robin

It is the simplest algorithm to distribute the load to the group of servers. All the requests are processed in a circular order irrespect of their priority, length etc as shown in figure2. It is best suited for the situations where all the servers are of equal capability in the terms of storage and processing power. There are variations of this algorithm in which weights can be assigned to each VM and then can wisely process the request.

Figure 2. Round Robin Load Balancing

B. Active Monitoring Load Balancing

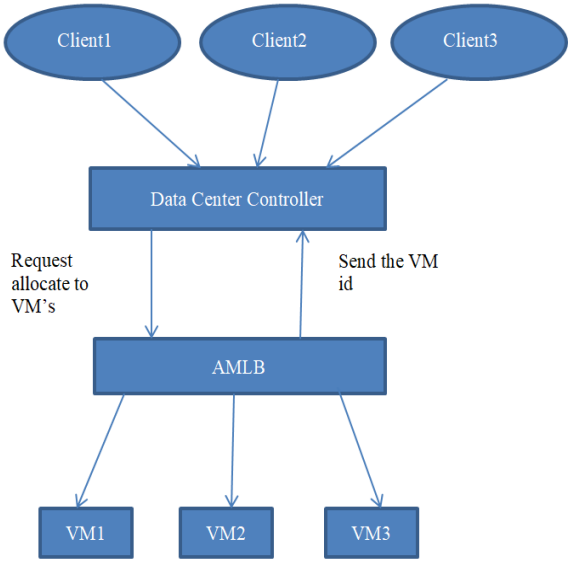

Figure 3: Active Monitoring Load Balancing

Equally distributed current execution (ESCE) load balancing is another name for this approach. It employs an active monitoring load balancer to evenly distribute load execution among multiple virtual machines as shown in Figure3. An index table of virtual machines and the quantity of allocations assigned to each virtual machine is maintained by the active monitoring load balancer (AMLB). A client sends a new request to Data Center Controller. When AMLB receives a request for a new VM from Data Center Controller, it parses the index table from top to bottom until the least loaded VM is discovered and returns its VM-ID to the data center controller. If more than one is discovered, AMLB will be notified. If more than one VM is detected, AMLB chooses the least loaded on a first-come, first-served (FCFS) basis. The Data Center then sends the VM designated by that id to the client. AMLB then updates the allocation table for that VM by increasing the allocation count by one. When a VM completed the processing of the assigned request satisfactorily, it sends a response to the Data Center Controller and alerts the AMLB for the VM de-allocation. The AMLB reduces the allocation count for that VM by one in the allocation table .

C. Raven Roosting LBT

The perching spot of raven birds fills in as a data community for broadcasting information about food supplies in their current circumstance. They normally check to verify whether there is enough food in their immediate surroundings, and if there is, they travel towards that spot in seek of food, if not, they look for food somewhere else. Similarly, each bird have their own specific aspect of food sources based on past experience, which serves as a decision element whether or not to approach the food source. Because of these properties, the raven roosting method is powerful enough to manage overload and underload problems in domains like cloud. The social nesting behaviour of ravens, which is to follow or not follow the ruler to obtain a big number of food sources, is replicated by tasks in the search for suitable virtual machines.

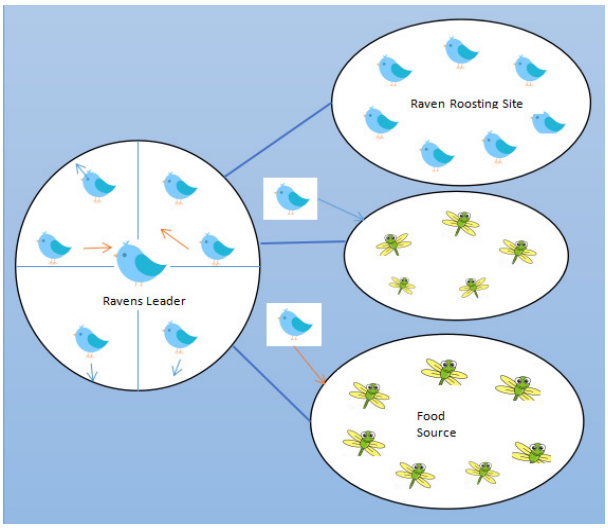

Fig 4. Raven Roosting Load Balancing

IV. SIMULATION STEP

The Cloud Analyst simulator is used to examine these algorithms. Cloud analyst is a graphical user interface application for modeling and analyzing large-scale cloud computing environments as shown in figure5. Furthermore, it allows the modeler to rerun the simulation while making changes to the parameters.

The Configure simulation, define internet characteristics, and run simulation menus are used to configure the

simulation process as a whole. The following evaluation parameters are used.

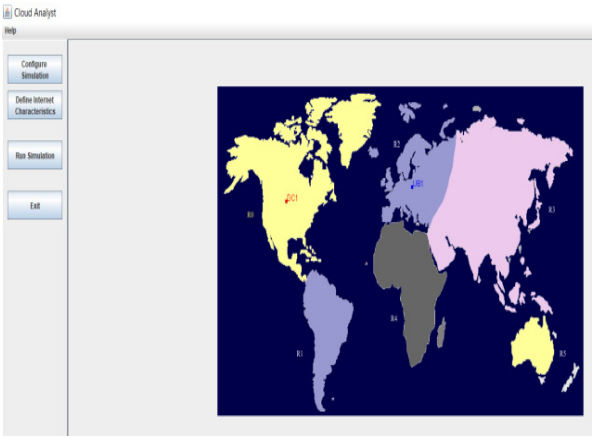

Figure5. GUI Interface of Cloud Analyst

A. Response Time Calculation

It is the amount of time it taken by a distributed system's load balancing algorithm to complete. This setting should be lowered for improved performance.

Response Time = FT - AT + TD

AT =Arrival Time of the user request,

FT =Completion time of the user request, and

TD = Transmission delay.

The, Transmission Delay can be calculated as:

TD = TL + TT

TL = Network Latency,

TT = Time taken to transfer the fixed size data of a single request (D) from the source to the destination.

B. Total Execution Time

The total time taken to assign jobs to random virtual machines, select the leader among the jobs, compute the step size for job movement, determine the leader's followers and unfollowers, and update the personal best of the job.

The tool has a function that allows us to switch algorithms based on our needs. The simulation setup and results are run for 60 minutes with varied numbers of users, three data centres (DC1, DC2, and DC3), and 75, 50, and 25 virtual machines (VMs) correspondingly as shown in figure6 and figure7. The remaining settings are set as stated in Table 1.

TABLE 1: Setting of Parameters

Parameter	Value passed
VM-image size	10000
VM-memory	1024MB
VM-Bandwidth	1000
Service Broker Policy	Opimise response time
Data center architecture	x86
Data center-OS	Linux
Data center-VMM	Xen
Data center-No of VMs	DC1 -75 DC2 -25 DC3-50
Data center-memory per machine	2GB
Data center-storage per machine	1GB
Data center-available bandwidth per machine	1000000
Data center-processor speed	1000
Data center-VM policy	Time shared
User grouping factor	1000
Request grouping factor	250
Executable instruction length	250

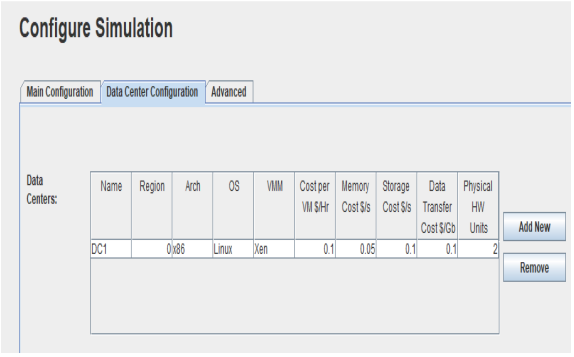

Figure6. Setting of parameters of Cloud Analyst

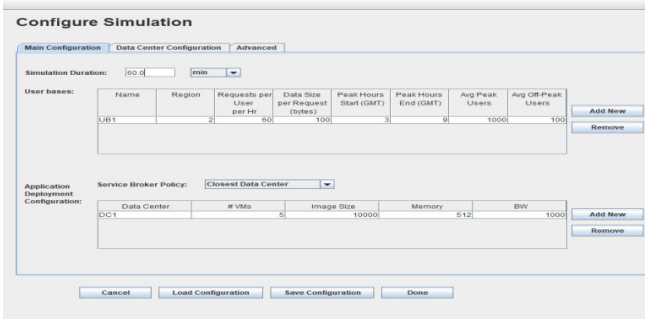

Figure7. parameters of Cloud Analyst

V. EXPERIMENTAL RESULTS

The results generated by cloud analyst after completing the simulation are provided in the tables below. We applied the above-described setup for each load balancing policy one at a time, and metrics like response time and request processing time in fulfilling the request were determined based on that. Tables 2, 3, and 4 show the overall response time computed by the cloud analyst for each loading policy, correspondingly. As can be observed from the table, the overall reaction times of the Round Robin and ESCEL policies are nearly identical, but the Throttled Policy's response time is relatively low when compared to the other two approaches.

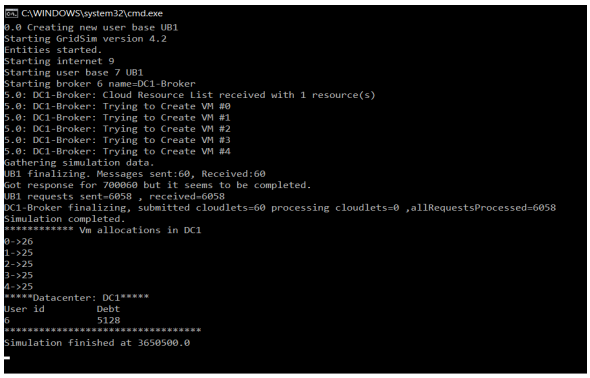

Figure 8. Virtual machine allocation

TABLE 2: Response time using Round Robin Policy
Case (I): Altering no. of user bases

User Bases (Ubs)	Time Complexity	Overall Response Time	Datacenter processing Time
5	Avg(ms)	300.12	0.35
	Min(ms)	229.60	0.02
	Max(ms)	373.67	0.65
10	Avg(ms)	300.04	0.35
	Min(ms)	229..62	0.02
	Max(ms)	393.11	0.80
15	Avg(ms)	299.89	0.35
	Min(ms)	229.63	0.02
	Max(ms)	387.12	0.82
20	Avg(ms)	299.89	0.35
	Min(ms)	229.63	0.02
	Max(ms)	387.12	0.82
25	Avg(ms)	299.94	0.36
	Min(ms)	225.11	0.01
	Max(ms)	393.12	0.97
30	Avg(ms)	299.96	0.35
	Min(ms)	211.62	0.01
	Max(ms)	387.12	1.06
35	Avg(ms)	300.03	0.36
	Min(ms)	208.61	0.01
	Max(ms)	387.11	1.09
40	Avg(ms)	300.03	0.36
	Min(ms)	208.61	0.01
	Max(ms)	396.12	1.22

Case (II): Altering no. of data centers

Data Centers	Time Complexity	Overall Response Time	Data center processing Time
5	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61
10	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61
15	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61
20	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61
25	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61
30	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61
35	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61
40	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61

TABLE 3: Response time using Active Monitoring Load Balancing Policy
Case (I): Altering no. of user bases

User Bases (Ubs)	Time complexity	Overall Response Time	Data center processing Time
5	Avg(ms)	300.12	0.35
	Min(ms)	229.60	0.02
	Max(ms)	373.67	0.64
10	Avg(ms)	300.04	0.35
	Min(ms)	229.62	0.02
	Max(ms)	393.11	0.70
15	Avg(ms)	299.89	0.35
	Min(ms)	229.63	0.02
	Max(ms)	387.12	0.82
20	Avg(ms)	299.97	0.35
	Min(ms)	223.61	0.02
	Max(ms)	379.61	0.88
25	Avg(ms)	299,93	0.35
	Min(ms)	208.61	0.01
	Max(ms)	388.61	0.89
30	Avg(ms)	299.97	0.35
	Min(ms)	222.11	0.02
	Max(ms)	388.62	0.92
35	Avg(ms)	300.01	0.36
	Min(ms)	207.14	0.01
	Max(ms)	384.45	1.05
40	Avg(ms)	300.03	0.36
	Min(ms)	208.61	0.01
	Max(ms)	387.11	1.06

Case (II): Altering no. of data centers

Data Centers	Time Complexity	Overall Response Time	Data center processing Time
5	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61
10	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61
15	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61
20	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61
25	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61
30	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61
35	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61
40	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61

TABLE 4: Response time using Throttled Policy
Case (I): Altering no. of user bases

User Bases (Ubs)	Time complexity	Overall Response Time	Data center processing Time
5	Avg(ms)	300.12	0.35
	Min(ms)	229.60	0.02
	Max(ms)	373.66	0.65
10	Avg(ms)	300.04	0.35
	Min(ms)	229.62	0.02
	Max(ms)	393.11	0.70
15	Avg(ms)	299.89	0.35
	Min(ms)	229.63	0.02
	Max(ms)	387.12	0.78
20	Avg(ms)	299.97	0.35
	Min(ms)	223.61	0.02
	Max(ms)	379.61	0.81
25	Avg(ms)	299,93	0.35
	Min(ms)	208.61	0.01
	Max(ms)	388.61	0.89
30	Avg(ms)	299.97	0.35
	Min(ms)	222.11	0.02
	Max(ms)	388.62	0.92
35	Avg(ms)	300.01	0.36
	Min(ms)	207.14	0.01
	Max(ms)	384.45	0.98
40	Avg(ms)	300.03	0.36
	Min(ms)	208.61	0.01
	Max(ms)	387.11	1.01

Case (II): Altering no. of data centers

Data Centers	Time Complexity	Overall Response Time	Data center processing Time
5	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61
10	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61
15	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61
20	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61
25	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61
30	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61
35	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61
40	Avg(ms)	300.06	0.34
	Min(ms)	237.06	0.02
	Max(ms)	369.12	0.61

VI. RESULT ANALYSIS

The overall response time and datacenter processing time of round-robin equally spread and Throttled load balancing technique is calculated by varying the number of data centers as 5,10,15,20,25,30,35,40 and calculated the complexity as average (ms), minimum (ms), and maximum(ms). We noticed that the dimensions of response time and data processing time are practically comparable in Round Robin and active monitoring load balancing policies. These parameters are marginally improved in the case of throttled load balancing. As a result, throttled load balancing appears more effective and efficient than the other two options.Though there is no significant difference in the response time and data processing time in all three algorithms even if we vary no. of user bases or no. of data centers. But there is a marginal difference between all these three algorithms, and by comparing these, we found out that the throttled algorithm is the best-suited algorithm. It is also quite clear from the table itself.

TABLE 5: Comparing all algorithms of Load Balancing Policy

User Bases (UBs)	Algorithm	Time Complexity	Overall Response Time	Data center processing Time
5	Round Robin	Avg(ms)	300.12	0.35
		Min(ms)	229.60	0.02
		Max(ms)	373.67	0.65
	Equally spread	Avg(ms)	300.12	0.35
		Min(ms)	229.60	0.02
		Max(ms)	373.67	0.64
	Throttled	Avg(ms)	300.12	0.35
		Min(ms)	229.60	0.02
		Max(ms)	373.66	0.65
10	Round Robin	Avg(ms)	300.04	0.35
		Min(ms)	229..62	0.02
		Max(ms)	393.11	0.80
	Equally spread	Avg(ms)	300.04	0.35
		Min(ms)	229.62	0.02
		Max(ms)	393.11	0.70
	Throttled	Avg(ms)	300.04	0.35
		Min(ms)	229.62	0.02
		Max(ms)	393.11	0.70
15	Round Robin	Avg(ms)	299.89	0.35
		Min(ms)	229.63	0.02
		Max(ms)	387.12	0.82
	Equally spread	Avg(ms)	299.89	0.35
		Min(ms)	229.63	0.02
		Max(ms)	387.12	0.82
	Throttled	Avg(ms)	299.89	0.35
		Min(ms)	229.63	0.02
		Max(ms)	387.12	0.78
20	Round Robin	Avg(ms)	299.89	0.35
		Min(ms)	229.63	0.02
		Max(ms)	387.12	0.82
	Equally spread	Avg(ms)	299.97	0.35
		Min(ms)	223.61	0.02
		Max(ms)	379.61	0.88
	Throttled	Avg(ms)	299.97	0.35
		Min(ms)	223.61	0.02
		Max(ms)	379.61	0.81
25	Round Robin	Avg(ms)	299.94	0.36
		Min(ms)	225.11	0.01
		Max(ms)	393.12	0.97
	Equally spread	Avg(ms)	299,93	0.35
		Min(ms)	208.61	0.01
		Max(ms)	388.61	0.89
	Throttled	Avg(ms)	299,93	0.35
		Min(ms)	208.61	0.01
		Max(ms)	388.61	0.89
30	Round Robin	Avg(ms)	299.96	0.35
		Min(ms)	211.62	0.01
		Max(ms)	387.12	1.06
	Equally spread	Avg(ms)	299.97	0.35
		Min(ms)	222.11	0.02
		Max(ms)	388.62	0.92
	Throttled	Avg(ms)	299.97	0.35
		Min(ms)	222.11	0.02
		Max(ms)	388.62	0.92

VII. CONCLUSION

Every engineer faces difficulty while developing solutions that can improve corporate performance in the cloud-based sector, such as reaction time and data transfer costs. The lack of efficient scheduling and load balancing resource allocation procedures in numerous strategies leads to higher operating costs and worse customer satisfaction. In a cloud-based environment, a more significant challenge in minimizing response time is generally noticed for each and every IT engineer to design products that may boost the efficiency of business performance and customer pleasure. With these considerations in mind, we simulated three possible load balancing algorithms for executing user requests in a cloud environment: round-robin, active monitoring, and throttled. Every algorithm is examined, and scheduling criteria such as average response time and data centre processing time are discovered.

In the case of Round Robin and active monitoring load balancing policies, we discovered that the parameters of response time and data processing time are nearly identical. However, in throttled load balancing, these parameters are marginally enhanced. As a result, we may conclude that throttled load balancing is more effective and efficient than the other two. Variating different parameters or inventing a new method for supplying cloud computing resources can improve the performance of the given algorithms. This approach could be tweaked and implemented for a real-time system in the future. We can expect a faster response time when we use evolutionary algorithms like PSO, ACO, and ABC instead of traditional algorithms.

REFERENCES

[1]. A. D. Josep, R. Katz, A. KonWinSKi, L. E. E. Gunho, D. Patterson and A. Rabkin, A view of cloud computing. Communications of the ACM, 53(2010), 130-143.

[2]. Mishra, Sambit Kumar, Bibhudatta Sahoo, and Priti Paramita Parida. "Load balancing in cloud computing: a big picture." Journal of King Saud University-Computer and Information Sciences 32, no. 2 (2020): 149-158.

[3]. Lavanya, V., M. Saravanan, and E. P. Sudhakar. "Self-Adaptive Load Balancing Using Live Migration of Virtual Machines in Cloud Environment." Webology 17, no. 2 (2020): 735-745.

[4]. Arulkumar, V., and N. Bhalaji. "Performance analysis of nature inspired load balancing algorithm in cloud environment." Journal of Ambient Intelligence and Humanized Computing 12, no. 3 (2021): 3735-3742.

[5]. Ebadifard, Fatemeh, Seyed Morteza Babamir, and Sedighe Barani. "A dynamic task scheduling algorithm improved by load balancing in cloud computing." In 2020 6th International Conference on Web Research (ICWR), pp. 177-183. IEEE, 2020.

[6]. Brabazon, Anthony, Wei Cui, and Michael O'Neill. "The raven roosting optimisation algorithm." Soft Computing 20, no. 2 (2016): 525-545.

[7]. Jena, U. K., P. K. Das, and M. R. Kabat. "Hybridization of meta-heuristic algorithm for load balancing in cloud computing environment." Journal of King Saud University-Computer and Information Sciences (2020).

[8]. Jin, Yu, Honggang Guo, Jianzhou Wang, and Aiyi Song. "A hybrid system based on LSTM for short-term power load forecasting." Energies 13, no. 23 (2020): 6241.

[9]. Rana, Nadim, and Muhammad Shafie Abd Latiff. "A cloud-based conceptual framework for multi-objective virtual machine scheduling using whale optimization algorithm." International Journal of Innovative Computing 8, no. 3 (2018).

[10]. Li, Xiaoxia. "A Study into Load Balancing in Cloud Computing Based on Whale Optimization Algorithm." CONVERTER 2021, no. 7 (2021): 180-189.

[11]. Abdel-Basset, Mohamed, Laila Abdle-Fatah, and Arun

Kumar Sangaiah. "An improved Lévy based whale optimization algorithm for bandwidth-efficient virtual machine placement in cloud computing environment." Cluster Computing 22, no. 4 (2019): 8319-8334.
[12]. Brabazon, Anthony, Wei Cui, and Michael O'Neill. "The raven roosting optimisation algorithm." Soft Computing 20, no. 2 (2016): 525-545.
[13]. Bhargavi, K., B. Sathish Babu, and Jeremy Pitt. "Performance Modeling of Load Balancing Techniques in Cloud: Some of the Recent Competitive Swarm Artificial Intelligence-based." Journal of Intelligent Systems 30, no. 1 (2021): 40-58.
[14]. Kumar, Rajeev, and Tanya Prashar. "Performance analysis of load balancing algorithms in cloud computing." International journal of computer Applications 120, no. 7 (2015).
[15]. www.degruyter.com
[16]. Sawsan Alshattnawi and Mohammad AL-Marie." Spider Monkey Optimization Algorithm for Load Balancing in Cloud Computing Environments
[17]. Azizkhan F Pathan Sneha.N Simulation of Load Balancing Algorithms using Cloud Analyst in International Journal of Engineering Research & Technology (IJERT)