ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 7, Issue 1, January 2019

Modified VM-Assign Load Balancing Algorithm of Cloud Computing in CloudSim

Prithpal Mohini Singh [1], Shaveta Angurala [2]

P.G. Student, Department of Computer Science, DAV Institute of Engineering and Technology, Jalandhar, India [1]

Assistant Professor, Department of Computer Science, DAV Institute of Engineering and Technology, Jalandhar, India[2]

ABSTRACT: In Cloud Computing one of the major challenges is efficient and effective utilization of resources to obtain quick response time and improve the performance of the Cloud Configuration. Different Load Balancing Techniques are developed for proper scheduling of available resources to effectively the overutilization or underutilization of resources. The paper presents and implements available load balancing algorithms with the proposed Modified Load Balancing Algorithm to improve the performance of the Cloud. CloudSim is implemented and results of the Load Balancing Algorithms are compared using parameters like Response Time and Data Processing Time The paper aims to provide the proposed Modified Load Balancing Algorithm a combination of three different methods, Batch Processing, Defined Value and Priority Algorithm, to balance the scheduling of load and for proper utilization of resources in Cloud Computing environment. The paper also provides comparative analysis of the proposed Modified Load Balancer with the existing Optimal Load Balancer Algorithm to analyse the results using Cloud-Analyst simulator using parameters like Response Time and Data Processing Time to get optimized results. It considers and implements four different case studies for comparing these parameters of the Load Balancing Algorithm to properly schedule available resources for better performance. The results showed optimized results on response time and Data Processing Time improving the performance and efficiency of the Cloud.

KEYWORDS: Cloud Computing, Scheduling, Load Balancing, Virtual Machines, Response Time, Utilization, Virtualization, Modified Load-Balancer, Data Processing Time, Cloud-Analyst, Simulation.

I. INTRODUCTION

The Cloud computing is defined as a model for enabling convenient, on-demand network access to a shared pool of configurable computing resource which includes networks, servers, storage, applications and services that can be rapidly provisioned and released with minimal management effort or service provider interaction. Cloud Computing, service model is defined as different types of applications provided by different servers across the cloud which is categorised as three types of service models, i) Software as a Service (SaaS), ii) Platform as a Service (PaaS), ii) Infrastructure as a Service (IaaS) and, private, public, hybrid and community are the four deployment models of Cloud Computing[1].

The process of distributing the total -load to the individual nodes/datacenters of the collective system to improve both resource utilization and job response time where resources are neither over-utilized or under-utilized , is called Load Balancing[2]. The main motive of load balancing is to achieve optimal resource utilization, Uniform distribution of load at different data centers, improving the overall performance of the system, higher user satisfaction, faster Response, and system stability[3][4]. The Load Balancing Algorithms are of two types, static and dynamic load balancing algorithms[5].

The main purpose of the proposed Modified Load Balancing algorithm is to minimize the response time and Data processing time to improve the performance and efficiency of performing tasks or to balance the load on the Cloud environment. The Load Balancing algorithms are based on two metrics[6] to analyze the performance and to optimize the resource utilization in the system i) Minimizing response time ii) minimizing the Data Processing time. The first metric focuses on the minimum, average and maximum time that is taken to respond to the task/job in the cloud system.

DOI: 10.15680/IJIRCCE.2019. 0701048

This parameter should be minimized for the better performance of the cloud. Second metric focuses on minimum, average and maximum time taken to process a task. This metric should be minimized for better efficiency of the system.

II. Related work

In [7] this paper, the authors proposed optimal-load balancing algorithm to check the overutilization or underutilization of resources or requests on cloud which are controlled and handled by VM to process the requests for better performance, throughput, response time. In [8] this paper, an algorithm for load balancing in cloud computing is implemented foraging the behavior of honey bees . When overloaded virtual machine is found, then the task is to be removed to the under loaded virtual machine to balance the load. Two situations arise i.e., either it finds the VM or it may not find the suitable VM. When it finds a proper VM, a positive signal is generated and when it fails to find suitable VM, a negative signal is generated. Also the task has to find best among the VMs when it finds more than one VM (set of VMs) which is done based on the QoS criteria called task priority. This means that the task finds the VM which has a less number of tasks with same kind of priority. When the task finds the suitable VM, the task is allocated to the respective VM found and the details are updated. In *[9]* this paper with the aid of genetic algorithm and fuzzy theory, present a hybrid job scheduling approach, which consider the load balancing of the system and reduces total execution time and execution cost. The main goal of this research is to assign the jobs to the resources with considering the VM MIPS and time span of jobs. The new algorithm assigns the jobs to the resources with considering the job length and resources capacities. Evaluate the performance of the approach with some famous cloud scheduling models. The result of the experiments shows the efficiency of the proposed approach in term of execution time, execution cost and average degree of imbalance. In [10] this paper, a method called “Central Load Balancer” which balances the load in cloud data centre. In large scale cloud computing environment this algorithm achieves well load balancing as compared to other load balancing algorithms. In this algorithm every request from user bases arrives at Data Centre Controller which are then processed by the Central Load Balancer which also maintains a table that consist of id, states and priority of VMs and finds out the highest priority VMs. The Central Load Balancer then checks the state of the highest priority VM. If its state is available, the id of that VM is returned to the Data Centre Controller. If the state is busy, then the next highest priority VM is chosen. After this the Data Centre Controller assigns the requests to that VMid that is provided by the Central Load Balancer. *In [11]* this paper, the Resource Scheduling is the process of mapping tasks to available resources on the basis of tasks characteristics and requirements. The received tasks are group on the basis of data and resources. Resource selection is done on the basis of its cost and turnaround times both using greedy approach and task selection on the basis of a priority. This way of resource selection and task selection gives better results over sequential scheduling. The available resources should be utilized efficiently without affection the service parameters of cloud. Main aim of this paper is to analyze the various scheduling algorithm and manage the resources which are precisely available at certain fixed times and for fifed intervals of time. Find the optimizes scheduling algorithm for resource so the cloud provider get benefits in term of efficient resource management which provide more resources to allocate without postponing or declining any user requests. Cloud users also get benefits in term of their monetary gains at each front. In [12] this paper, an algorithm called “Honey Bee Behaviour Inspired Load Balancing” which works well by distributing the load equally across virtual machines. This algorithm achieves load balancing so effectively that it reduces the amount of waiting time of tasks in the queue. In [13] this paper, an algorithm in which the fault tolerance in cloud computing using dynamic load balancing technique is taken into consideration. In this method CPU utilization is checked and the load is balanced according to the utilization of the CPU. With the help of load balancer, the tasks are shifted from the virtual machines with high CPU utilization to the virtual machines with less CPU utilization.

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

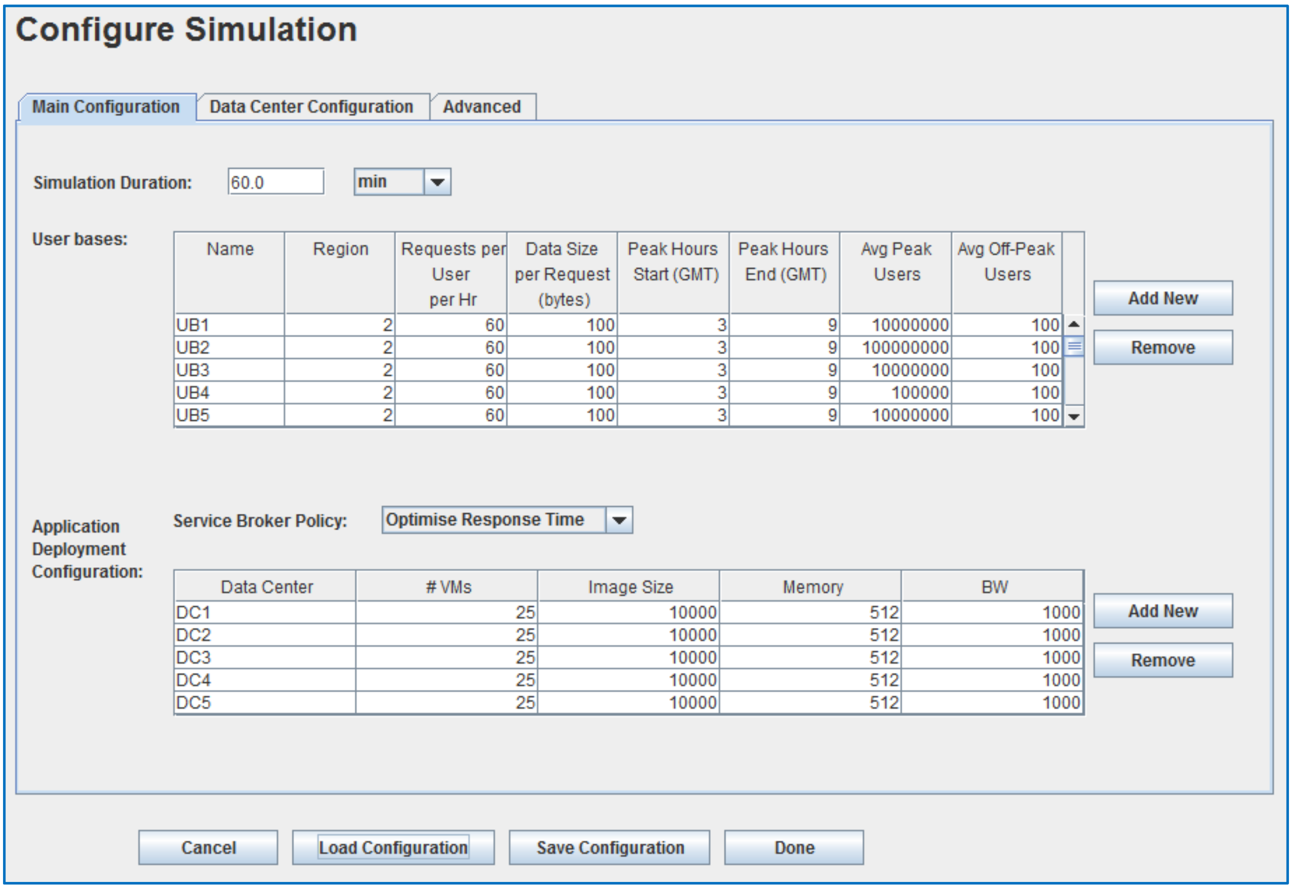
Website: www.ijircce.com

Vol. 7, Issue 1, January 2019

III. Proposed Algorithm

A. *Design Considerations:*

- For Main Configurations, the Simulation time is 60 mins.
- User-Bases up to 5 are set for different geographical locations, Request Per-User-Per Hour value is set up to 60, and Data-Size-Per Request (in bytes) is 100.
- Application Service Center describes two parameters as Service Broker Policy: Response Time, Data Processing Time.
- For Data Centers, Case Studies from VM to 75 are considered; image size 10000, memory 512 and BW 1000 are set as default values.
- Figure 1 shows the GUI used to configure the specifications for User-bases, and Application Deployment Configuration, where the Data centers, Virtual Machines and other entities can be specified and defined as per simulation requirements.



Configure Simulation

Simulation Duration: 60.0 min

User bases:

Name	Region	Requests per User per Hr	Data Size per Request (bytes)	Peak Hours Start (GMT)	Peak Hours End (GMT)	Avg Peak Users	Avg Off-Peak Users
UB1	2	60	100	3	9	10000000	100
UB2	2	60	100	3	9	100000000	100
UB3	2	60	100	3	9	10000000	100
UB4	2	60	100	3	9	100000	100
UB5	2	60	100	3	9	10000000	100

Application Deployment Configuration:

Service Broker Policy: Optimise Response Time

Data Center	# VMs	Image Size	Memory	BW
DC1	25	10000	512	1000
DC2	25	10000	512	1000
DC3	25	10000	512	1000
DC4	25	10000	512	1000
DC5	25	10000	512	1000

Figure 1 Configure Simulation of Cloud Analyst Simulator

B. *Description of the Proposed Algorithm:*

Aim of the proposed Modified VM-Assign Load Balancing algorithm is to improve scheduling of user requests thereby optimizing the performance and efficiency by considering and comparing parameters like response time and data processing time with the Optimal Load Balancing Algorithm hence reducing the cost and maximizing the performance of the Cloud Environment. The Proposed Algorithm Modified-VM Load Balancer Algorithm combines batch processing, Priority Algorithm and Defined Value to get faster response time, lower data processing time, reduced cost and efficient performance. It overcomes the flaw of not assigning the next/ incoming user-request to the VM which was allocated in the previous assignment as batch is used to allocate requests which saves time of comparison of individual VMs apart from checking for their availability, current allocation time and whether it was used in the previous request or not. In this case the response time of the algorithm could degrade and hence reduce the performance of the Cloud. The Modified Load Balancer firstly checks for the number of user-requests to be served and allocated though a function called Batch, which specifies the batch size and the maximum capacity of load allocation on individual VMs/Resources based on number of jobs/user-requests received. The Algorithm then calculates the current allocation of individual VMs and selects the maximum value of current allocation count. The defined value is calculated for every VM by deducting the Current allocation size from the maximum capacity that can be allocated to every VM (maximum capacity of individual VMs is decided by the batch function depending on the number of user-requests received). The defined value is then compared with the Maximum current allocation value for every VM, to set priority on individual VMs. The highest defined value means lesser user-requests are allocated to the VM; hence higher priority, so the higher defined values get high priorities to which batch of user-requests/jobs can be allocated. After allocating the VM, it updates the allocation table and calculates the defined values again for comparison and prioritizes the VMs for task/job allocation. Instead of parsing the table, checking the availability status, and the last assigned VM every for single user-request as specified in Optimal Scheduling. The proposed Algorithm parses Allocation table for batch of user-requests, thereby reducing the number of comparisons and certain conditions which leads to faster response time and better performance of the system.

Step 1: Calculating the Data Load:

Suppose there are ‘n’ numbers of user-base requests in different regions that need to be scheduled

$$\text{R}= \{R_1, R_2,\ R_{3,}\ R_4\ , \dots\dots,\ R_n\} \qquad (1)$$

If there are ‘m’ set of Virtual Machines at every Data Center or Region

$$\text{V}= \{VM_1, VM_2,\ VM_{3,}\ VM_4\ , \dots\dots,\ VM_m\} \qquad (2)$$

The Data Load can be defined by $\text{L} = \{\text{V}\} * \{\text{R}\}$

$$= \{VR_1, VR_2,\ VR_{3,}\ VR_4\ , \dots\dots,\ VR_n\} \qquad (3)$$

A method/function f(R) needs to be defined so as to map the set of request or load can be distributed to Virtual Machines available at different geographical locations to get optimized result for balancing load by considering

$$VR_1 \approx VR_2 \approx VR_{3,} \approx\ VR_4\ \ \approx .., \approx\ VR_n\} \qquad (4)$$

Let us use σ to reflect the time needed for executing task *Lo* on the Virtual Machine *Vi*, time required to process all requests/tasks at V_i will be $t_i =\ \sum_{o\,\in f(R_i\,(\,i=1\,to\,n))} \sigma_0$ (5)

The numbers of Virtual Machines are given by ‘m’. When m = 1, only one Virtual Machine is used to execute the user-based requests or tasks. The time taken to complete serve all the requests on given machine is the sum total of the tasks that are executed by the virtual machine: $\text{T} = \sum \sigma_0$ (o = 1 to n) (6)

For Multiple virtual Machines, m>1, the user requests can be shared and executed by different virtual machines depending on the availability of the server in different regions.

$$\text{T} = \sum max_{i=1\ to\ n}\, t_i \qquad (7)$$

Step 2 Selection Criteria:

The Proposed Technique to balance load in the Cloud by scheduling requests/jobs to get optimized response time as compared to the previous Optimal Load Balancing Technique, implements three techniques to get better results and faster response time than the base algorithm. The Proposed Method implements Batch Processing, Defined Value and Priority Scheduling to allocate the VMs to the user requests or tasks. The Batch Processing works based on the number of user-requests received. If the number of user-requests are less than or equal to 100, then the batch size to be allocated to the VM is 5 and the maximum capacity of the load that can be allotted to the VMs is 90. To generalize the batch processing function formulae:

Batch (If UserReqN <= 100, then Batch_size = 5 AND MaxAllocCapacity = 90.

…………………………………………………………………………

If UserReqN <= N, then Batch_size = .05N AND MaxAllocCapacity = .9N)

Step 3: Assigning the Load to VMs at different Geographical Locations.

For Every VM calculate the Defined Value as, Defined_Value = MaxAllocCapacity – CurrAllocVal.

The Modified Load Balancer selects the MaxCurrAllocValue for comparison with the Defined Values.

Assigns Priority to every VM based on Defined value, highest_priority to the Highest (Defined_Value), so on and the lowest_priority to the least Defined_Value. The Load Balancer then selects the VM and allocates Batch to it.

IV. PSEUDO CODE

Input: No of incoming user-requests R1, R2 . . . . Rn.

Available The number of VMs; VM1, VM2 . . . . . . VMn.

Output: All incoming user-requests R1, R2 . . . . Rn are allocated according to the batch function and priority is assigned according to DefVal of individual resources among the available VMs; VM1, VM2 . . . . . . VMn.

```
Step 1: Initially all the VM's have 0 allocations.
               Initialize VMs = 0;
Step 2: Modified VM-assign load balancer maintains the Allocation table of VMs which has no. of requests
          currently allocated to each VM, DefinedValue, Priority of VMs.
            VM(id, State, CurrentAllocationValue, DefinedValue, Priority);
Step 3: When user requests arrive at the data-center, it passes the jobs over to the load balancer.
Step 4: The function Batch is defined,
             Batch (UserRequestN, NB, MaxAllocCapacity)
            (If (UserRequestN <= 100) then NB = 5 AND    MaxAllocCapacity = 90.
            If (UserRequestN <=10000) then NB = 500 AND MaxAllocCapacity = 9000.
            .......... ………………………………………………………
            If (UserRequestN<=n) then NB = .05n  AND MaxAllocCapacity = .9n;)
Step 5: Calculate the CurrentAllocationValue on every VM.
Step 6: For every VM/Resource Defined Value is calculated by subtracting the Current Allocation value of
         user requests from the Maximum Allocation value, which specifies the maximum load/maximum
         number of user request that can be allocated to a VM and is specified by the batch function
                        DefValue = MaxAllocVal – CurrentAllocationValue.
            a. For Every VM, Compare the DefinedValue with the Max(CurrentAllocationValue) and
                  Check and parse table for the condition and value to assign priority to every VM;
                        For (Max(DefinedValue)= priority; priority++):
                             Select Max(DefinedValue); assign VM( priority).
                 Cases:
                  1.    When Maximum Current Allocation is greater than the Defined Value,
                                  Max(CurrentAllocationValue) > DefinedValue = Assign(lowest_priority).
                  2.    When Maximum Current Allocation is less than the Defined Value,
                                    Max(CurrentAllocationValue) < DefinedValue = Assign(highest_priority).
```

```
                    Return to Step a.
            b. Allocate the batch() to the VM( highest _priority).
                    Update VM(CurrentAllocationValue).
            Return to Step 5.
Step 7: A Response is received at the Data-Center after VM has finished the user-request/job.
Step 8: The data center notifies the Modified VM-assign load balancer for the VM de-allocation and
        updates the table;
Step 9:  Return to Step 2.
```

V. SIMULATION RESULTS

The simulation studies involve Case Studies of VMs from 5 to 75 at each Data processing Center located at different Geographical locations. The proposed Modified Load Balancing algorithm is implemented with CloudSim(CloudAnalyst). Proposed algorithm is compared by two parameters Response Time and Data Processing Time. Response Time is the amount of time that is taken by a particular load balancing algorithm to response a task in a system. This parameter should be minimized for better performance of a system. Data Processing Time is the amount of time actually needed to process a task. The performance of the Load Balancing algorithms is measured by considering the parameters like response time and data processing time.

Figure 2 displays Cloud-Analyst Simulator which shows Data Centres, Regions, and User-Base at different geographical Locations. Table 1 shows the Comparative Response time (ms) for the Load Balancing Algorithms when different case scenarios are used to study the impact of increasing the number of VMs at different data centres. Table 2 shows the Comparative data processing time (ms) for the Load Balancing Algorithms for different case studies. Table 3 displays the Comparative average response time of the load balancing algorithms. Figure 3 shows graphical comparison of the algorithms for parameters average response time and average data processing time. Table 4 shows the percentage improvement/ optimization in average response time for the load balancing algorithms. Table 5 show the results of different case studies of VM from 5 up to 75 for average data processing time then calculates the difference in the data processing time for the two algorithms.

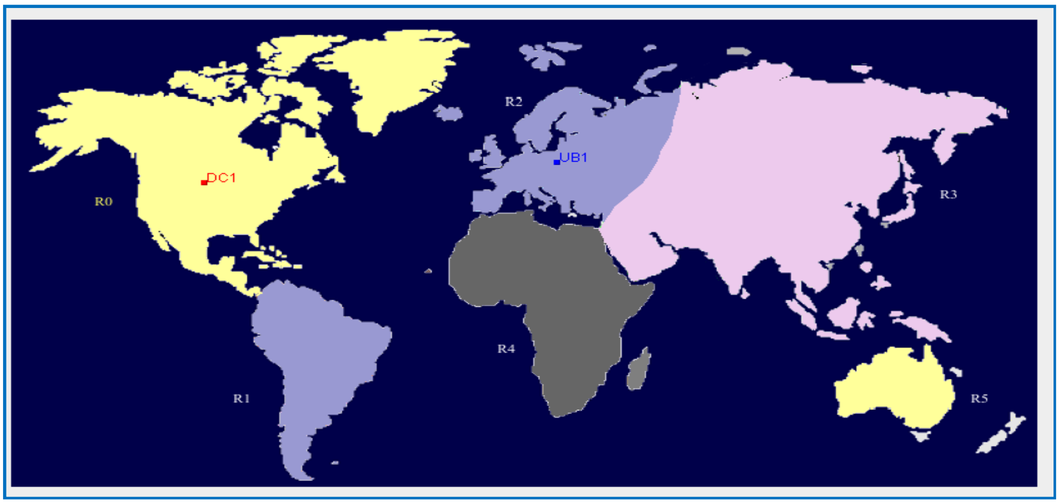


Figure 2 Cloud Analyst Simulator: Region.

Table 1 displays the average, minimum and maximum Response time(in ms) for Different Case-Studies when VM 5 to 75 are considered for the Proposed Algorithm, Modified VM-Load Balancer and the optimized results that are obtained when compared to the Optimal Load Balancing algorithm for different case studies. Faster the response time , better is the performance and of the efficiency of the algorithm. For every case study, the proposed algorithm shows faster response time.

Table 1 Comparison Table for Response Time (ms) for Different Case Studies

CASE STUDY	ALGORITHM	Avg(ms)	Min (ms)	Max (ms)
CASE 1: VMs=5	Optimal Load	385.34	37.61	18053.02
	Modified VM-Load Balancer	**262.14**	**37.61**	**16398.55**
CASE 2: VMs=25	Optimal Load	1137.26	65.25	36691.79
	Modified VM-Load Balancer	**673.45**	**50.25**	**30823.81**
CASE 3: VMs=50	Optimal Load	387.84	37.62	17752.26
	Modified VM-Load Balancer	**260.14**	**37.61**	**16395.77**
CASE 4: VMs=75	Optimal Load	355.35	36.86	18053.50
	Modified VM-Load Balancer	**244.08**	**35.12**	**16392.25**

Data Processing time is the amount of time actually needed to process a task. Considering the same case studies form VM 5 to 75, for optimizing the data processing time of the existing Optimal Load Balancer and comparing and analysis results with the Proposed Modified Load Balancer Technique. The results clearly display optimized results where average, minimum and maximum time(in ms) for data processing time are better than the previous algorithm, Optimal Load balancing algorithm.

Table 2 Comparison Table for Data Processing Time for Different Case Studies

CASE STUDY	ALGORITHM	Avg (ms)	Min (ms)	Max (ms)
CASE 1: VMS=5	Optimal Load	273.12	0.01	17999.00
	Modified VM-Load Balancer	**150.15**	**0.01**	**16336.05**
CASE 2: VMS=25	Optimal Load	968.08	0.00	36639.75
	Modified VM-Load Balancer	**504.15**	**0**	**30765.56**
CASE 3: VMS=50	Optimal Load	276.01	0.02	17697.75
	Modified VM-Load Balancer	**148.81**	**0.02**	**16337.52**
CASE 4: VMS=75	Optimal Load	249.75	0.02	17999.25
	Modified VM-Load Balancer	**139.64**	**0.02**	**16337.50**

The Case study Configuration increases the number of user-requests or tasks and the number of virtual machines or resources in the system. In Case of five VMs, the Average Data processing time is reduced from 273 to 150 approximately concluding that the proposed algorithm has improved the data processing time by 45%. For twenty-five VMs, the Average Data processing time is reduced from 968 to 504 approximately concluding that the proposed algorithm has improved the data processing time by 49%.

Table 3 Load Balancing Algorithms Comparison

ALGORITHM	AVG RESPONSE TIME (ms)	AVG DATA PROCESSING TIME (ms)
OPTIMAL LOAD	2264	1766
MODIFIED LOAD	***1439***	***941***

Figure 3 displays the results that are obtained from table 3 where average response time and average data processing time are compared and difference in the optimized results are displayed and compared using graphs for algorithms.

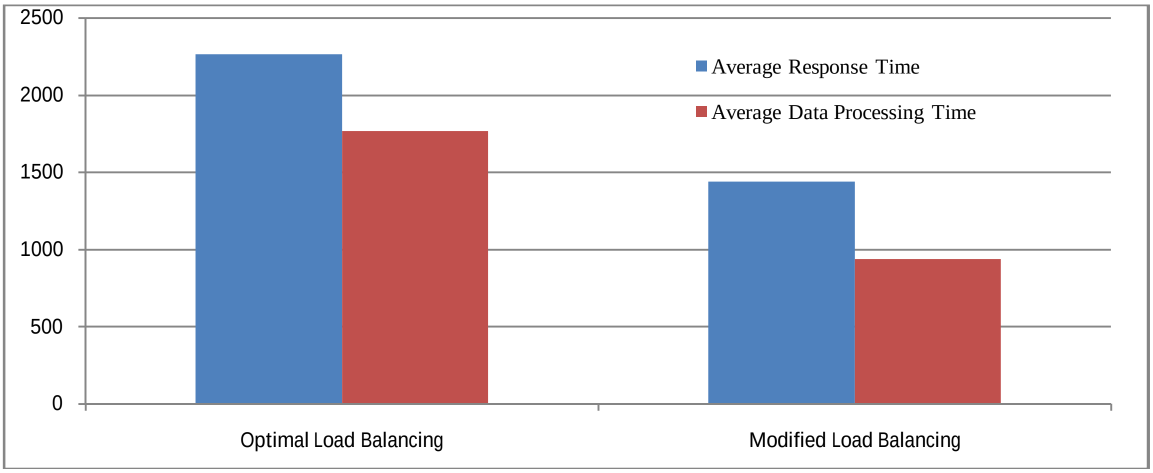


Figure 3.Comparison Optimal Load Balancing Algorithm and Modified Load Balancing Algorithm

Table 4 displays the improvement in average response time when the proposed and existing algorithms are compared for the case studies from VM 5 to 75. In every case, Modified VM-Load algorithm displays faster response time. The difference in the change shows the optimized percentage of the algorithms.

Table 4 Percentage Improvement/Optimization in Response Time

No. of VMs	ALGORITHM	Avg(ms)	% Optimization
CASE 1: VMs=5	**Optimal Load**	385.34	**31.9%**
	Modified VM-Load	**262.14**	
CASE 2: VMs=25	**Optimal Load**	1137.26	**40.8%**
	Modified VM-Load	**673.45**	
CASE 3: VMs=50	**Optimal Load**	387.84	**32.8%**
	Modified VM-Load	**260.14**	
CASE 4: VMs=75	**Optimal Load**	355.35	**31.2%**
	Modified VM-Load	**244.08**	

Table 5 displays case studies from VM 5 to 75 and compares the average Data processing time for the two algorithms to exhibit which algorithm faster processing and the percentage optimization shows by how much the time improved and hence the efficiency of the cloud. The Optimal Load shows more data processing time in every case and the proposed algorithm shows faster and better average data processing time and the percentage improvement in overall time.

Table 5 Percentage Comparison in Data Processing Time for different Case Studies

CASE STUDY	ALGORITHM	Avg (ms)	% Improvement
CASE 1: VMS=5	**Optimal Load**	273.12	**45%**
	Modified VM-Load	**150.15**	
CASE 2: VMS=25	**Optimal Load**	968.08	**47.9%**
	Modified VM-Load	**504.15**	
CASE 3: VMS=50	**Optimal Load**	276.01	**46.3%**
	Modified VM-Load	**148.81**	
CASE 4: VMS=75	**Optimal Load**	249.75	**44.1%**
	Modified VM-Load	**139.64**	

VI. CONCLUSION AND FUTURE WORK

The simulation results showed that the proposed algorithm, Modified Load Balancing Algorithm performs better than the Optimal Load balancing Algorithm with parameters like Response time and Data Processing time displaying optimized results and better performance. Each case study takes different values of VM to study the impact of algorithm by increasing load and resources at different geographical regions. The impact of the proposed algorithm can be further studied by increasing the number of user requests or tasks to be performed. Further case studies can be conducted by increasing the user requests and number of VM allotted at different data centers (scaling up the configuration of the cloud environment), using parameters like response time and data processing time for the load balancing algorithms. The performance of the proposed algorithm Modified VM-assign Load Balancing Algorithm is analyzed by two parameters, in future with some modifications in pseudo algorithm, and more parameters like fault tolerance, CPU utilization, resource utilizations can be considered to check and upgrade the performance of the proposed algorithm and can be further compared with other load balancing algorithms. Optimized Load Balancing helps in proper utilization of resources so they are neither over-utilized nor underutilized thereby minimizing resource consumption.

REFERENCES

1.“NIST Cloud Computing Program–NCCP”,Nov 2010. [Online]. Available:https://www.nist.gov/programs-projects/nist-cloud-computing-program-nccp.

2. V.Suresh Kumar, “Resource Scheduling for Load Balancing Using Ant Colony Optimisation”, Vol 117, No. 22, 2017.

3. Foram F. Kherani, and Prof. Jignesh Vania, “Load Balancing in Cloud Computing", IJEDR, Vol 2, Issue 1, 2014.

4. D. Chitra Devi, and V.Rhymend Uthariaraj, “Load Balancing in Cloud Computing Environment Using Improved Weighted Round Robin Algorithm for Nonpremembtive Tasks, Scientific World Journal, Vol 2016.

5. Abhijit Aditya, Uddalak Chatterjee, and Snehasis Gupta, “ A Comparative Study of Different static and dynamic Load Balancing Algorithm in Cloud Computing with special Emphasis on Time Factor”, International Journal of Current Engineering and Technology, Vol. 5, No.3, 2015

6. Nitin Kumar Misra, and Nishchol Mishra , “Load Balancing Techniques: Need, Objectives, and Major Challenges in Cloud Computing- A systematic review”, International Journal of Computer Applications, Vol 131, No. 18, 2015.

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 7, Issue 1, January 2019

7. Shridhar. G. Donamal, "Optimal Load-Balancing in Cloud Computing by efficient utilization of virtual resources", IEEE, 2014 Sixth International Conference on Communication Systems and Networks (COMSNETS), Jan 2014.
8. Obaid Bin Hassan et al, "Optimal Load Balancing of Cloudlets using Honey Bee Behaviour Load Balancing Algorithm", International Journal of Advance Research in Computer Science and Management Studies, Vol 3, Issue 3, 2015.
9. Saeed Javanmardi, MohammadShojafar, Danilo Amendola, and Nicola Cordeschi, " Hybrid Job scheduling Algorithm for Cloud computing Environment", Proceedings of the Fifth International Conference on Innovations in Bio-Inspired Computing and Applications IBICA 2014, Vol 3, pp 43-52, 2014.
10. Gulshan Soni and Mala Kalra," A Novel Approach For Load Balancing In Cloud Data Center", 2014 IEEE international advance computing conference (IACC), pp 807-812, 2014.
11. Hitesh A. Ravani, Hitesh A. Bheda, and Vruda J Patel, "Genetic Algorithm Based Resource Scheduling Technique in Cloud Computing", International Journal of Advance Research of Computer Science and management Studies, Volume 1, Issue 7, 2013.
12. Dhinesh Babu L.D., and P. Venkata Krishna, "Honey Bee Behavior Inspired Load Balancing Of Tasks In Cloud Computing Environments", Applied Soft Computing Journal, Vol. 13, Issue 5, pp 2292-2303, 2013.
13. Argha Roy, and Diptam Dutta, "Dynamic Load Balancing: Improve Efficiency In Cloud Computing", International Journal Of Emerging Research In Management &Technology, Vol. 2, Issue 4, pp 78-82, 2013.