Available Online at www.ijcsmc.com

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. 15, Issue. 1, January 2026, pg.137 – 159

An Efficient Hybrid Load Balancing Algorithm for Reduce Response Time in Cloud Computing

[1]Zahra Mohammed Elngomi; [2]Professor. Saif El-Din Fattoh Osman; [3]Dr. Fatima Alzhra Mohammed Said

[1]PhD. Program in Computer Science, National Ribat University, Khartoum.
[2]Full professor Head of Information Security Department at National Ribat University, Khartoum.
[3]Associate Professor in Information Technology collage at National Ribat University, Khartoum.
[1]Email: Zohraelngomi@gmail.com

DOI: https://doi.org/10.47760/ijcsmc.2026.v15i01.010

ABSTRACT: Recently Cloud computing has become popular term due to its attractive features. In the cloud computing environment, with the ease of access to Internet the users and their requirements are increasing day by day. Load balancing consideration as one of the most importance challenge in cloud computing. Several algorithms have been developed to dealing with load balancing challenge but still there some issues did not solve which impacted in the performance. In this work we are proposing a hybrid approach to enhance performance for the cloud system based on the concepts of improve throttled and weight round robin (WRR). The proposed hybrid algorithm (PTMA) will be simulated in cloud analyst environment. The results evaluated using two factors there are response time and throughput.

Keywords: Cloud Computing, Load Balancing, Improve Throttled, Weight Round Robin (WRR), Performance, Response Time, Throughput, Cloud Analyst

1. Introduction:

1.1 Cloud Computing:

The NIST defines Cloud computing is "a computing model used everywhere and provides convenient, on-demand access to a shared pool of computing resources such as networks, servers, storage, applications, etc." [1-3]. these resources can be dynamically allocated and released with minimal management effort or service provider interaction. Cloud computing has become one of the technologies which was adopted by both industry and academia because it provides flexible and efficient method to store and access files. The service provider's is responsible about software installations, upgrades and maintenance in Cloud Computing environment, also the load balancer is responsible for keeping load balancing on the cloud all the time by insure, the users did not need to concern about this issues [4]. Despite the benefits of cloud computing which provides storage of data in very lower cost and available for all time over the internet, there are critical issues facing us when we deal with it such as insecurity, load balancing, fault tolerance ability, scheduling of resource, Qos management and Scale. These challenges have negatively affect performance [5]

1.2 Load Balancing:

Load balance is the process of distributed or re-distributed work load among various nodes of a server in the cloud, this process can be achieve in two ways, allocation requests at first time or migration task from node to node to insure that there no node over load while other node under node which its lead to keep balance overall system and improve the performance. Load balancing are necessary to improve the performance over all entire system by achieving optimal resource utilization, maximum throughput and minimum response time. Efficient load balancing depended on efficient scheduling algorithms. [6-7-8].

There are two concepts of load balancing algorithms in cloud computing and that are mention below:

1.3 Types of Load Balancing Algorithms:

1.3.1 Static Load Balancing:

Static load balancing mainly based on the information about average of the system workload it does not take the actual current system status into account [2-3]. In static algorithms takes decision about load balancing at compile time.

1.3.2 Dynamic Load Balancing:

Dynamic algorithm depends on the current state of system and take the change of attribute in consideration. Dynamic algorithm works based on the different properties of the nodes such as network bandwidth and capabilities.

Dynamic load balancing algorithms distributed the work load at the run time. The decision of the balancing load is taken depended on the current status of the system.[8]

1.4 Existing Load Balancing Algorithms:

1.4.1Throttled Load Balancing Algorithm:

It is better in virtual machines. Load balancer keeps the list of all virtual machines in the system in indexed table according to the state(busy/idle). When it is receiving a request, it searches about a suitable virtual machine in indexing table depend on state. The load balancer scans the index table from top until the first available VM was found or the index table was scanned fully, if it was found, then the job will be assigned to that machine else return -1 and the request is put in queue. Load balancer updates the indexing table after each process allocation and de-allocation. It does not take the current load of VM while allocating a request. [10-16]

1.4.1.1 The sequence of steps:

Step 1. Throttled Load Balancer execution load balancing by update and maintain an index table contains the status information (available '0' or not available '1') of all VMs. At start, all VM at the status is available '0'.

Step 2. Data Center Controller received a new request.

Step 3: Data Center Controller query to Throttled Load Balancer for the new task.

Step 4: Throttled Load Balancer checks VMs table from top to down, determined the first VM is available.

If found VM:

-Throttled Load Balancer sends the ID of VM to Data Center Controller.

-The Data Center Controller sends a request to the VM specified by that ID.

-Data Center Controller notifies the Throttled Load Balancer a new allocation.

-Throttled Load Balancer updates the index and waits for new requests from the Data Center Controller.

If not found VM:

-Throttled Load Balancer will return a value of -1 to the Data Center Controller.

-The Data Center Controller arranges the request.

Step 5: As for the VM, after processing the request and the Data Center Controller receives a response, it will notify to Throttled Load Balancer is finished.

Step 6: If there are multiple requests, the Data Center Controller repeats **Step 3** with the next index and the process is repeated until the index table size is empty.

Note: This algorithm optimizes the response time than the Round Robin algorithm. But the limitation is to detect the VM is available '0' with the index table size increase. [10-11]

1.4.1.2 The Throttled Load Balancing Algorithm Flowcharts:

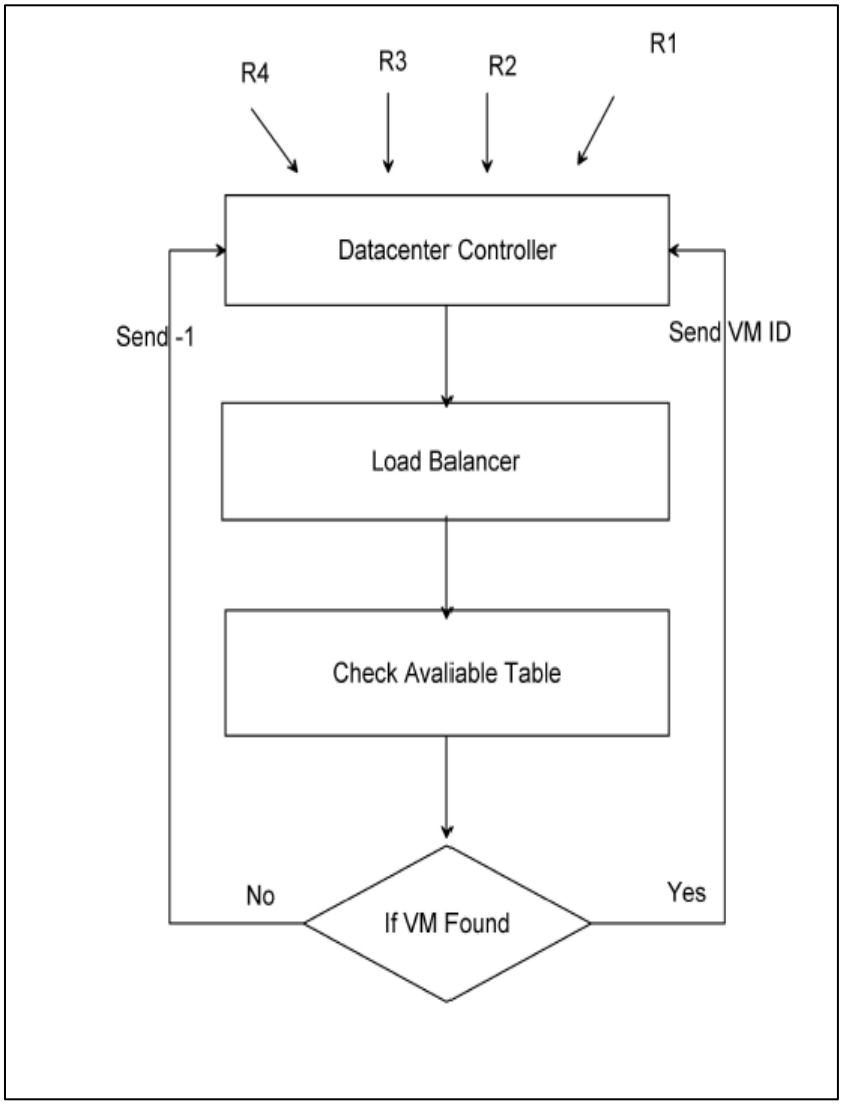


Figure (1): Throttled Load Balancing Algorithm Allocated Flowchart

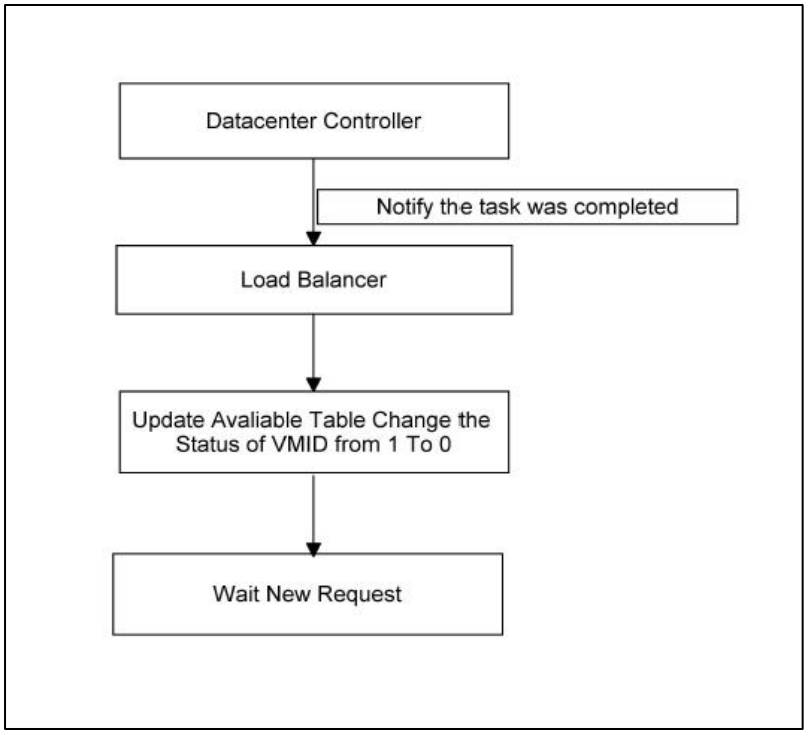


Figure (2): Throttled Load Balancing Algorithm De-Allocated Flowchart

1.4.2 Weighted Round Robin(WRR):

It is the newer version of Round Robin in which each VM has weight according to its ability so that if one VM can handle double load than other, then one machine with weight 2 can handle two tasks while the VM with weight 1 handle one task. The execution time of individual requests is not taking in consideration. [22]

There are two ways to serve the request in this algorithm mention as its shown below:

1.4.2.1 Classical WRR:

In this method the scheduler selects the queue with highest weight and When a queue is selected, the scheduler will send packets of this queue up to the task counter equal the queue weight or the end of queue. [15]

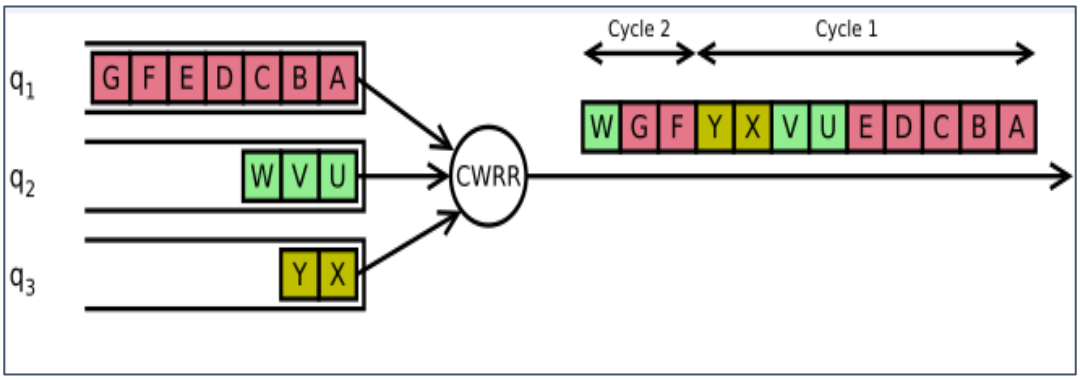


Figure (3): Classical WRR

1.4.2.2 Interleaved WRR:

In this method, the scheduler determines the max of weight of the queues then split each cycle to the number round equal the max weight and give each queue a chance to send task to serve until the tasks counter become greater than max weight or queue was empty, then select new max. [15]

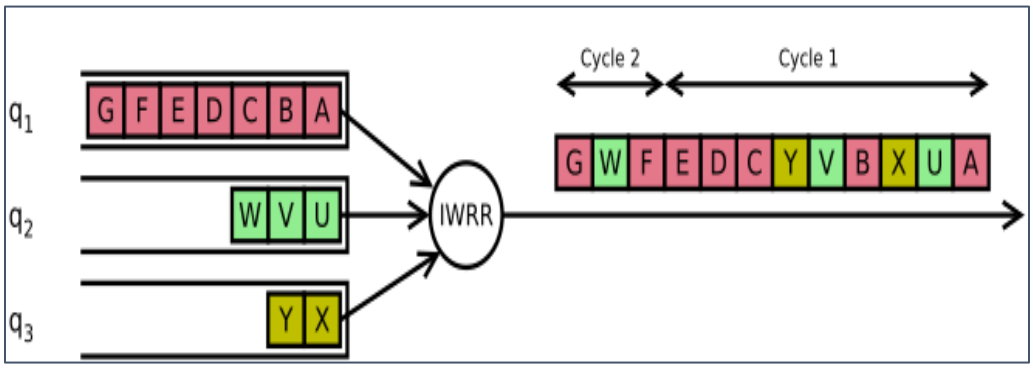


Figure (4): Interleaved WRR

2. Related Works:

In this paper [1]Modified Throttled Load Balancing algorithm proposed, this algorithm split the process of client obtain services from the service provider into three steps in flexibility way. Firstly, it initialization stage. this stage icalculate the expected response time of each VM. secondly is detect the suitable VM. Finally, is return suitable VM ID to the data center. The expected response time of VM computed by using the following formula:

Response Time = Fint − Arrt + TDelay

where,

Arrt → Arrival time of user request

Fint → Finish time of user request.

TDelay → Transmission delay which is computed using the below formula:

TDelay = Tlatency + Ttransfer

where,

TDelay → Transmission delay

Tlatency → Network latency

Ttransfer → The amount of time required for transmission of the data size of the single request (D) from a source to destination is computed by using the below formula:

Ttransfer = D/Bwperuser

where,

Bw → Bandwidth per user is computed using below formula

Bwperuser = Bwtotal/Nr

where,

Bwtotal → Total available bandwidth

Nrl → Number of user requests which are currently in transmission. [4]
By using the above formulae, the response time of the virtual machines is computed and then an efficient virtual machine can be obtained among them. The drawback of this algorithm that the selection of suitable VM was inaccuracy, because predicated response time may be different in real run time.

In this paper [4] a hybrid load balancing algorithm was introduced, based on both random and greedy algorithms to enhance the performance and efficiency in a heterogeneous cloud computing by optimization resources utilization. The proposed hybrid algorithm takes consideration of current resource data and the CPU capacity factor. The mechanism work of hybrid is: Firstly, VMs are distributed over host depended on the of the host specification. The CPU capacities of the host determine the numbers of VMs created on that host. Secondly created index table to register the current loads for each VM. When the hybrid load balancer receives a request from data center to serve users ,it will randomly choose a group of nodes (VM), then testing the current load for each VM and select a VM with least current Disk loads, return the Disk id to Data Center. The data center will have allocated the request to the VM, and update the current load index table. Finally, when the VM finishes processing the request, the data center will be notifying the load balancer to update its current load value. Despite the achievement of goal proposed algorithm by reducing the response time and processing time, there is a drawback on this algorithm, that the migration was inefficiency.

In this paper [9], the proposed algorithm improved throttled algorithm (ITA) original throttled algorithm will enhance response time, request processing time, and reduce the cost of the Data center. With this algorithm, we select a virtual machine with least load from the available table. The least load represented the number of task which it works in this VM now. This approach helps load balancer get fast processing time and minimize idle resources The drawback of this algorithm is that the load balancer choose the VM depended on only the number of task working now did not take other factors in consideration that can be impact in the performance such as processor power, execution time...etc.

in this paper [10], An Improved throttle algorithm has been proposed, aim to enhancement the allocation process. priority is determined to assign each VM and kept in index table. It is taken capacity of VM, active allocated task count and size in consideration and using it is to calculate the priority, when a request is arriving to the load balancer choose highest among the available set of VM, also it will be start searching VM in VM allocation table from the next to the last allocated VM. By this way, the proposed algorithm improve performance by make allocation process more efficiency and reduce the response time. The drawback of this algorithm is that it may cause overheads due to the arithmetic operations used to calculate the priority.

In this paper [12], an improved algorithm has been proposed from the quantitative algorithm. This algorithm aims to reduce the response time and processing time by reducing the time to search for the appropriate virtual machine by dividing the table containing the data of the virtual machines into two tables (Available, Busy), the opposite of what is done in the original algorithm. Experiments have proven that the proposed algorithm offers a better response time. Although the results obtained from the proposed algorithm prove it has met the goals, there are drawback of this algorithm that its focused only on the reduce the search time for suitable VM, did not takes Characteristics of hosts (execution time, task size,CPU power...etc.) or task in consideration.

In this paper [16] Give expanding view of many load balancing schemes have been reviewed along with the in depth understanding. The paper main focus on the modified and improved versions of throttled load balancing algorithm. The paper is only concentrated on the working of these variants of throttled load balancing algorithm.

In this paper [18], an improved version of the weighted round robin load balancing algorithm (EWRR) was proposed, aims to improve the average response time and average processing time of the system in the cloud environment which lead to utilization of resource efficiently. For manage the allocation process efficiently the proposed algorithm will take account of some factors which impact direct in the performance such as cloudlet size, expected execution time of tasks, runtime properties. The process of allocation and migration of tasks is carried out based on an equation to calculate the waiting time by using characteristics of the task, Where the waiting time is equal to the sum of the factorial of a number Length of Cloudlet and Execution time of VM, where the execution time is calculated based on Cloudlet and its equal Cloudlet Size divided by of VM power. Load balancer create a table contented the waiting time for each task. Mapping is completed by insert a task with upper waiting time afterward recognizes the best appropriate virtual machine for execution of the task in the virtual machine lists and assigning a target task onto the identified VM based on the waiting time. Load balancing is done by re arrangement of the order so depends on the load of the VM calculate the waiting time. The drawback of this paper, that the selection process of the suitable VM depend on the expected execution time of the task on this VM, and sometime at run time of system, tasks may take more or longer time of execution than the initially expected time.

3. Methodology:

In this section, we will present the proposed hybrid algorithm which depended on both throttled and weight round robin algorithms. The main aim of this paper is to improve the performance by reducing average response time which lead to increasing the throughput by using effectively utilizing VMs processor power taken in account. Thus, this leads to attainment of the optimum utilization of resources.

3.1 Proposed Load Balance Algorithm steps:

The Throttled Algorithm performance was considered good. But, in throttled algorithm methodology only one task can be assigned to a VM. This means that if the task assigned is not utilizing the VM to its maximum capacity then it cannot be a perfect solution. If the task assigned needs heavy load and was not available In VM can lead to overload. Therefore, in this paper, an algorithm is proposed which is a new version of the Throttled algorithm called the Proposed Throttled Modified Algorithm (PTMA), aims to enhance the performance of throttled algorithm by optimizing utilization of resources. [2]

Step 1. The PTMA Load Balancer performs load balancing by updating, maintaining two index tables.

- Available Index: content VM Id and Weight of VM (weight means the VM had ability to handle numbers of request equal the weight no).
- Busy Index: content VM Id, Weight of VM and Task counter (task counter mean the number of task that this VM is processing now).

At the beginning, all VMs are updated in the "Available Index" table and the "Busy Index" table is empty.

Task counter = Weight.

Step 2: The Data Center Controller receives a new request.

Step 3: Data Center Controller query to PTMA Load Balancer for the new task.

Step 4: PTMA Load Balancer will be checked the available index table, determined the VM has highest weight.

If found VM:

- PTMA Load Balancer sends the ID of VM to Data Center Controller.
- The Data Center Controller sends a request to the VM specified by that ID.
- Data Center Controller notifies the PTMA Load Balancer a new allocation.
- PTMA Load Balancer updates the index by decrement the specified VM weight

by 1.

If weight = 0.

- The PTMA Load Balancer sends VM Id and switches VM Id and it is weight to the Busy Index.
- The PTMA Load Balancer wait for the new request from Data Center Controller

If weight >=1

- The PTMA Load Balancer update VM available index table, by new VM weight.
- wait for the new request from Data Center Controller

Note: If there more than one VM with the same Weight the PTLM select any one randomly.

If not found VM:

- PTMA load balancer will be checked the busy index table, determined the VM has highest different between weight and Task counter value.

If found VM:

- PTMA load balancer sends the ID of VM to Data Center Controller.
- The Data Center Controller sends a request to the VM specified by that ID.
- Data Center Controller notifies the PTMA Load Balancer a new allocation.
- PTMA load balancer updates the busy table by decrement the specified VM task counter by -1.

Note: If there more than one VM that the different between Weight and task counter give the same value the PTMP select any one randomly.

If not found VM:

- PTMA load balancer will return a value of -1 to the Data Center Controller.
- The Data Center Controller arranges the request.

Step5: As for the VM, after processing the request and the Data Center Controller receives a response, it will notify to TLM Load Balancer is stopped.

- PTMA load balancer will decrement that specific VM Id task counter by 1.
- Check the task counter:

If Task counter = 0.

- The PTMA Load Balancer update available index table, switch VM Id and is weight from busy index table to available Index table.
- wait for the new request from Data Center Controller

If Task counter >=1

- The PTMA Load Balancer wait for the new request from Data Center Controller repeat step 4.

We are assuming update of “available index” and “Busy index “table happen after any allocate and de-allocate process**.**

Step 6: If there are multiple requests, the Data Center Controller Repeats **Step 3**

3.2 Proposed Procedures Flowcharts:

There are two main functions in the PTMA the first one is when the DC query the LB search about the best suitable VM to allocate the request to served. Second one is when the DC notify the LB that the process was finishing and the VM become available. the flowcharts below explain the procedure of those functions:

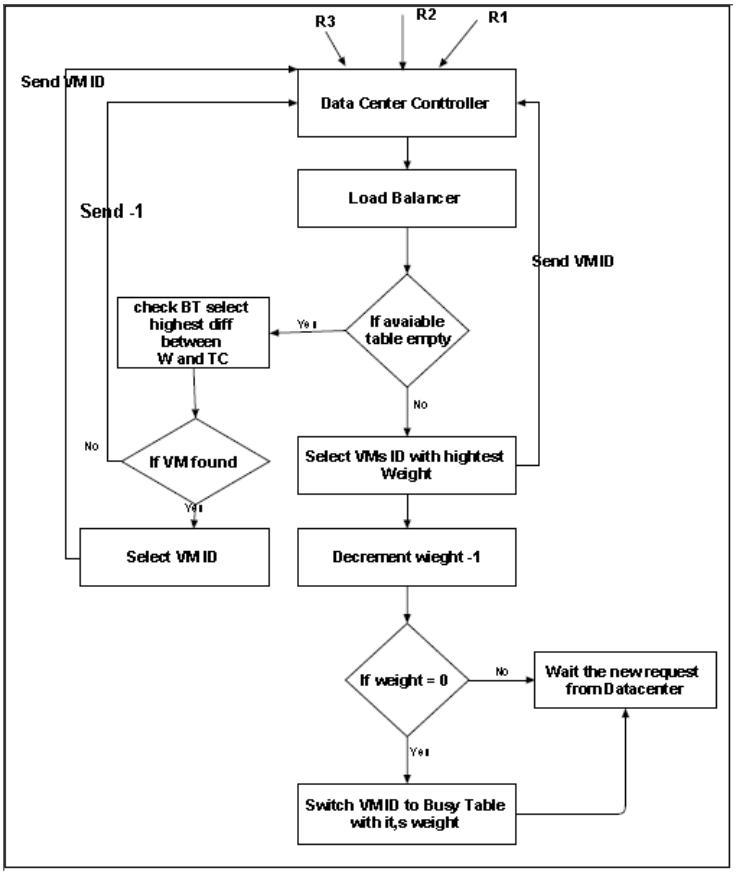


Figure (5): PTMAAllocatedFlowchart

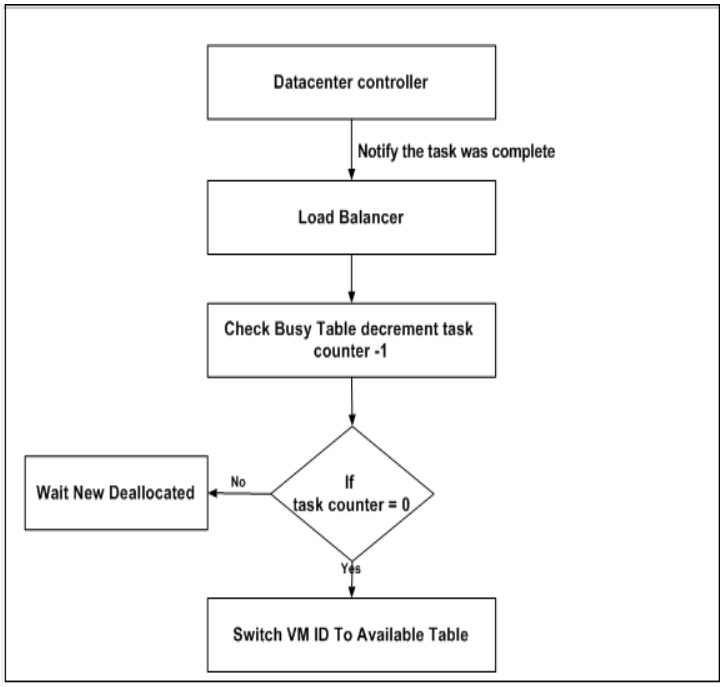


Figure (6): PTMA De-allocate Flowchart

3.6.3 Proposed Method Choose VM:

In the proposed algorithm, we have adopted a new method for distributing tasks to the virtual machines based on the current weight, in this method was taking current weight VM into account. The weight has decrement after each cycle. This method was a hybrid between CWRR, IWRR in Weight Round Robin algorithm. We assume this method is fair more than previous methods use in WRR[19].

Example:

Consider a system with three queues. Assumethat: q1 weight =5, q2 weight =2, q3 weight =3. Consider a situation where they are 7 packets in the first queue, A, B, C, D, E, F, G, 3 in the second queue, U, V, W and 2 in the third queue X, Y. Assume that there is no more packet arrival. The figure below explains the distributed of tasks in:

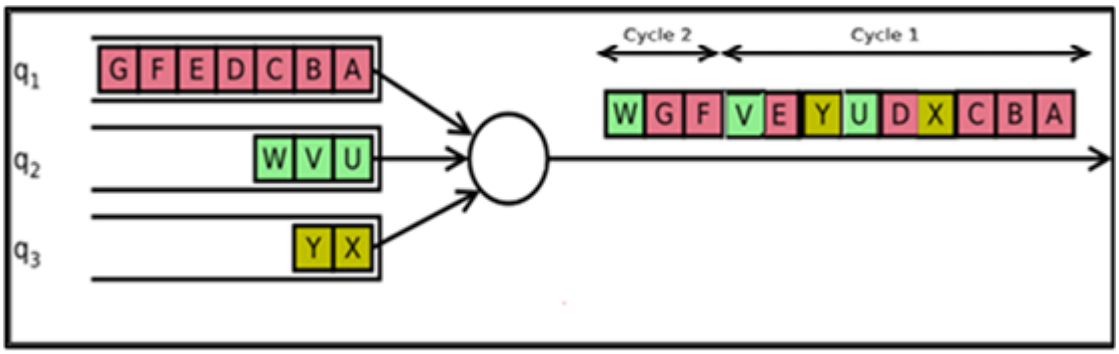


Figure (7): Proposed Method Select VM

proposed In the first cycle, the scheduler first selects q1(sincew1 = 5)and transmits the packets A ,B,Cahead of queue because it's weight higher until it's w =2, then it selects the second queue q3 (sine w=3)and transmits the one packet at head of queue X , then transfer to q1 (since w 2 = 2) and transmits the one packet at head of queue D, then return to q2(sine w=2)) and transmits the one packet at head queue U,then return to q3 (sine w=2) and transmits the last packet at a head queue Y ,then return to q1 (since w=1)and transmits the one packet E, return to q2 (sine w=1) and transmits the packet at a head queue V , then the weight become q1 (since w= 0) seem q2(sine w =0),q3 (sine **w=1**) but it is empty , then the first cycle was finished and second cycle start with the beginning weight for the queues, so transmits the last packet at head of q1 F and G,then send the last packet W from q2.

Note: If there two VM has the same weight the VM is select randomly to serve the task.

4. Experiments and Results:

This section gave account of the experiments and results. Cloud Analyst simulator was used to make comparison between the proposed algorithm and the current load balance algorithms. Cloud analyst is one of the efficiently simulators in world using in testing environment to make experiments and evaluate the performance of load balancing algorithms. Is one of attractive simulators because it has a graphical user interface, which it maddens easy to use tool, it also has

ability to perform simulations and repeat experiments continuously by restore the save configuration parameters easily, it quickly in very less time without effort.[3]

4.1 Experimental Setup:

The user base configuration parameters configured as it mention in the tables (1). Several simulator configuration such as (configuration of users, configuration of data centers, configuration of VMs) identify in beginning of each experiment, other point has to be taken into account that, there are other significance attributes play a prominent amount of role while execution process is induced such as service broker policy, the experiment was conducted and repeated by using optimize response time service broker. We studied the problem without the effect of network delay and latency, first time we conduct the experiments in heterogeneous environment of hosts, where each machine has different number of CPUs and speed, then run the experiment in homogenious environment of hosts to evaluate the effect of CPU factor power. [4]

The performance of algorithms evaluated under obvious monitor process such that the algorithm can be changed whenever there is particular requirement or need.

Finally, we are conducting experiments on PTMA, Throttled, Round Robin and ESCE algorithms with the same parameters in the same environment.

We assume that the Duration of simulate is 60 minutes.[20-21]

Following are the details of the main factors and policies adopted for implementation of the experiments:

Table (1): User Base Configure

User Grouping Factor	Request Grouping Factor	Executable Instruction Length
10	**10**	**100**

4.2 Experiments Implementing:

4.2.1 Experiment No1:

In this experiment cloud analyst had been configured using 4 Data center distributed around 3 reigns, each datacenter established 5 virtual machines (Virtual machines with 512 MB of memory, image size 10000 and 1000MB of bandwidth available). Datacenter physically has (204800 Mb of memory running on physical processors capable of speeds of 100MIPS) with different number of CPUs, there are 7 User Base distributed that corresponds to 4 regions with a specific peak hour start from 3 to 9 GM and avg peak user was 1000.The user request per hour is 60 request,

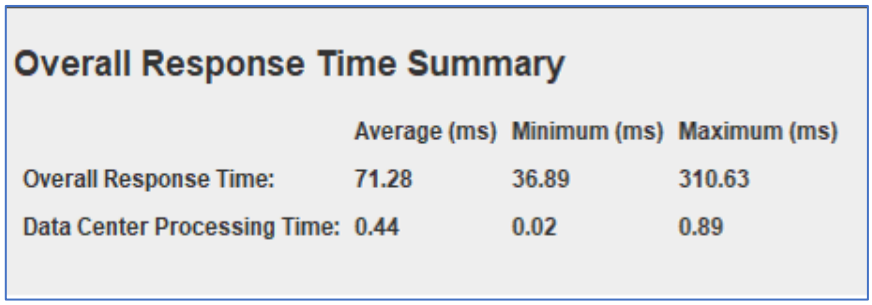
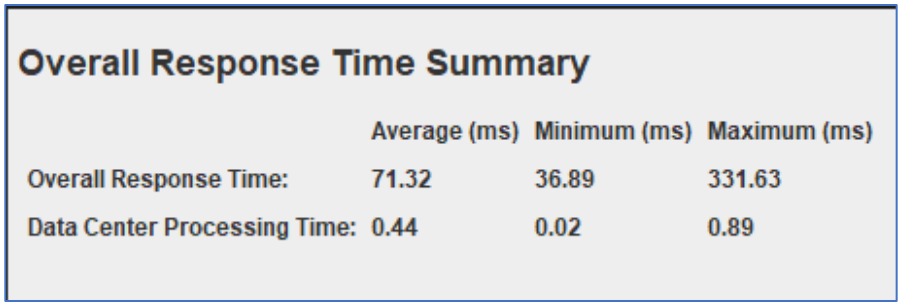


Figure (8) : PTMA –Overall Response Time - Experiment 1



Overall Response Time Summary

	Average (ms)	Minimum (ms)	Maximum (ms)
Overall Response Time:	71.32	36.89	331.63
Data Center Processing Time:	0.44	0.02	0.89

Figure (9): Throttled –Overall Response Time - Experiment 1

Algorithm	Response Time			Processing Time	Throughput
	Avg	Min	Max		
RR	71.36	36.89	331.63	0.44	42,697
ESCE	71.29	36.89	331.63	0.44	42,697
Throttle	71.32	36.89	331.63	0.44	42,697
PTMA	71.28	36.89	331.63	0.44	42,697

Table (2): Comparison Among all load balancing algorithms -Experiment 1

User Base	Algorithms			
	PTMA	Throttled	RR	ESCE
UB1	199.49	199.684	199.675	199.60
UB2	50.07	50.094	50.128	50.04
UB3	50.11	50.26	50.194	50.06
UB4	49.91	50.885	50.333	49.96
UB5	50.08	50.991	50.075	50.06
UB6	50.08	50.069	50.103	50.11
UB7	50.26	50.329	50.025	50.25

Table (3): Comparison Among all load balancing algorithms (Average response time(ms)) - Experiment 1

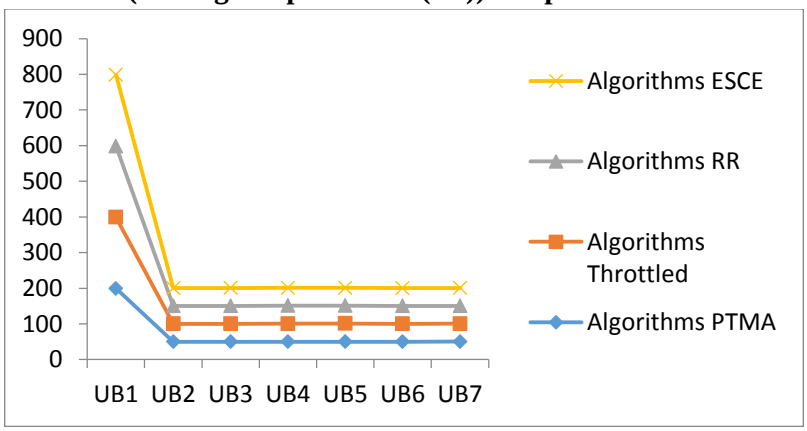


Figure (10) : Analysis showing seven user bases vs. average response time (ms)- Experiment No 1

In this experiment, we found that the overall response time improved in the proposed algorithm by (0.02) ms comparing with the throttled algorithm and improved in by (0.04) comparing with the worst result, as well as the throughputs was (42,697) remained constant, although the overall response time is less in the proposed algorithm than the original algorithm, also there was no effect on processing time and the cost of the two algorithms.

4.2.2 Experiment No2:

In this experiment cloud analyst had been configured using 4 Data center distributed around 4 reigns, each datacenter established 5 virtual machines (Virtual machines with 512MB of RAM memory, image size 10000 and 1000MB of bandwidth available). Datacenter physically has (204800 Mb of memory running on physical processors capable of speeds of 100MIPS) with different number of CPUs there are 6 User Base distributed that corresponds to 5 regions with a specific peak hour start from 3 to 9 GM and average peak user was 1000.The user request per hour is 60 request.

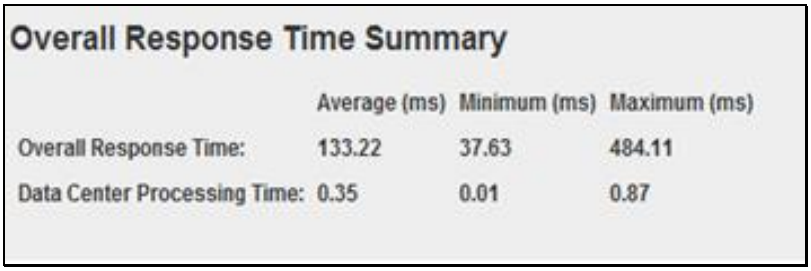


Figure (11): PTMA–Overall Response Time - Experiment 2

Figure (12) : Throttled – Overall Response Time – Experiment 2

Algorithm	Response Time			Throughput
	Avg	Min	Max	
RR	133.23	37.63	484.11	36,513
ESCE	133.27	37.63	484.11	36,513
Throttle	133.27	37.63	484.11	36,513
PTMA	133.22	37.63	484.11	36,513

Table (4): Comparison Among all load balancing algorithms -Experiment 2

User Base	Algorithms			
	PTMA	Throttled	RR	ESCE
UB1	200.23	199.688	200.48	200.47
UB2	49.92	49.96	49.96	49.97
UB3	49.94	50.045	49.95	49.97
UB4	50.27	50.164	50.2	50.18
UB5	50.18	50.284	50.16	50.11
UB6	400.06	400.773	399.88	400.19

Table (5) : Comparison Among all load balancing algorithms (Average response time (ms)) - Experiment 2

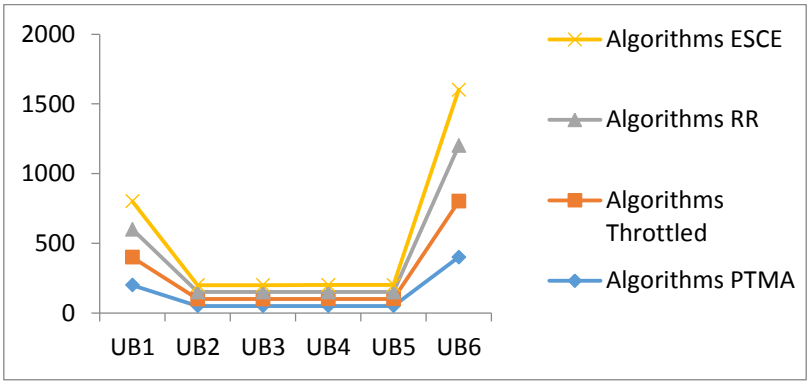


Figure (13) : Analysis showing six user bases vs. average response time (ms) - Experiment 2

In this experiment, we found that the overall response time improved in the proposed algorithm by (0.05) ms comparing with the throttled algorithm, as well as the throughputs was (36,513) remained constant, although the overall response time is less in the proposed algorithm than the original algorithm, Also there was no effect on processing time and the cost of the two algorithms

4.2.3 Experiment No3:

In this experiment cloud analyst had been configured using 4 Data center distributed around 4 reigns, each datacenter established 5 virtual machines (Virtual machines with 512MB of RAM memory, image size 10000 and 1000MB of bandwidth available). Datacenter physically has (204800 Mb of memory running on physical processors capable of speeds of 100MIPS) with different numb8er of CPUs there are 7 User Base distributed that corresponds to 5 regions with a specific peak hour start from 3 to 9 GM and avg peak user was 1000.The user request per hour is 60 request,

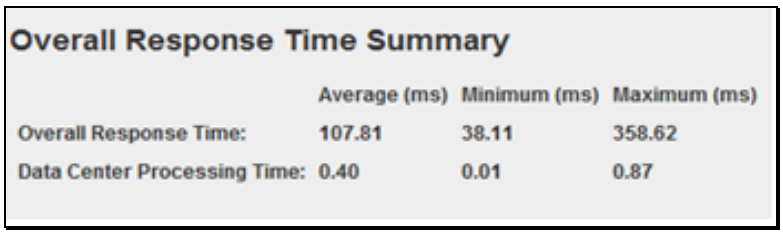


Figure (14): PTMA–Overall Response Time - Experiment 3

Figure (15) : Throttled–Overall Response Time - Experiment 3

Algorithm	Response Time			Throughput
	Avg	Min	Max	
RR	107.84	38.11	358.62	42,697
ESCE	107.83	38.11	358.62	42,697
Throttle	107.87	38.11	358.62	42,697
PTMA	107.81	38.11	358.62	42,697

Table (6) : Comparison Among all load balancing algorithms -Experiment 3

User Base	Algorithms			
	PTMA	Throttled	RR	ESCE
UB1	49.939	49.909	49.93	49.93
UB2	50.147	50.093	50.1	50.09
UB3	50.054	50.094	50.09	50.12
UB4	49.879	49.907	49.89	49.88
UB5	299.37	299.684	299.53	299.45
UB6	199.89	199.684	199.92	199.92
UB7	50.236	50.955	50.21	50.23

Table (7) : Comparison Among all load balancing algorithms (Average response time (ms)) - Experiment 3

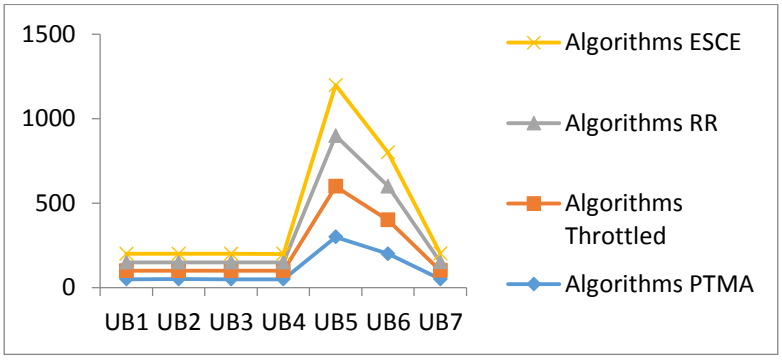


Figure (16) : Analysis showing ten user bases vs. average response time (ms) – Experiment 3

In this experiment, we found that the overall response time improved in the proposed algorithm by (0.06) ms comparing with the throttled algorithm, as well as the throughputs was (42,697) remained constant, although the overall response time is less in the proposed algorithm than the

original algorithm, Also there was no effect on processing time and the cost of the two algorithms.

4.2.4 Experiment No4:

In this experiment cloud analyst had been configured using 4 Data center distributed around 4 reigns, each datacenter established 5 virtual machines (Virtual machines with 512MB of RAM memory, image size 10000 and 1000MB of bandwidth available). Datacenter physically has (204800 Mb of memory running on physical processors capable of speeds of 100MIPS) with same number of CPUs, there are 6 User Base distributed that corresponds to 4 regions with a specific peak hour start from 3 to 9 GM and avg peak user was 1000.The user request per hour is 60 request,

Figure (17) : PTMA–Overall Response Time - Experiment 4

Figure (18) : Throttled–Overall Response Time - Experiment 4

Algorithm	Response Time			Throughput
	Avg	Min	Max	
RR	165.39	37.63	607.62	36,513
ESCE	165.38	37.63	607.62	36,513
Throttle	165.39	37.63	607.62	36,513
PTMA	165.35	37.63	607.62	36,513

Table (8) : Comparison Among all load balancing algorithms -Experiment 4

User Base	Algorithms			
	PTMA	Throttled	RR	ESCE
UB1	499.1	499.37	499.172	499.18
UB2	49.96	49.94	49.905	49.97
UB3	300.12	300.14	300.324	300.28
UB4	50.16	50.19	50.166	50.14
UB5	50.3	50.25	50.305	50.24
UB6	50.2	50.2	50.252	50.26

Table (9) : Comparison Among all load balancing algorithms (Average response time (ms)) - Experiment 4

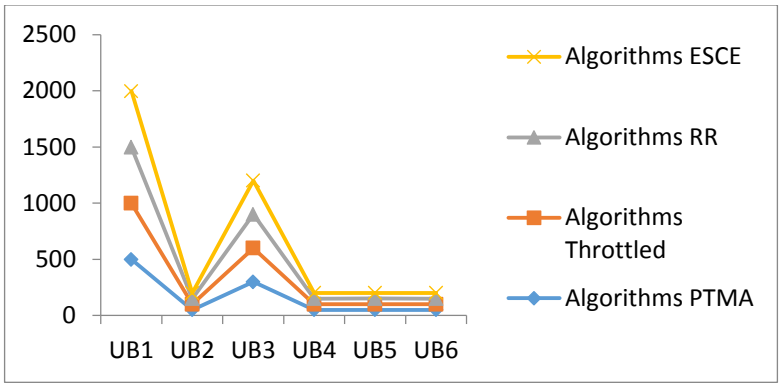


Figure (19) : Analysis showing ten user bases vs. average response time (ms) – Experiment 4

In this experiment, we found that the overall response time improved in the proposed algorithm by (0.04) ms comparing with the throttled algorithm, as well as the throughputs was (36,513) remained constant, although the overall response time is less in the proposed algorithm than the original algorithm, Also there was no effect on processing time and the cost of the two algorithms.

4.2.5 Experiment No5:

In this experiment cloud analyst had been configured using 4 Data center distributed around 4 reigns, each datacenter established 5 virtual machines (Virtual machines with 512MB of RAM memory, image size 10000 and 1000MB of bandwidth available). Datacenter physically has (204800 Mb of memory running on physical processors capable of speeds of 100MIPS) with same number of CPUs there are 7 User Base distributed that corresponds to 5 regions with a specific peak hour start from 3 to 9 GM and avg peak user was 1000.The user request per hour is 60 request.

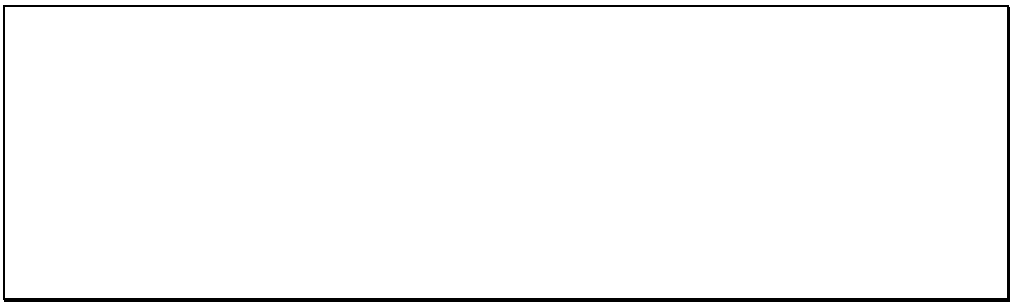


Figure (20): PTMA–Overall Response Time - Experiment 5

Overall Response Time Summary

	Average (ms)	Minimum (ms)	Maximum (ms)
Overall Response Time:	114.74	38.11	250.12
Data Center Processing Time:	0.31	0.01	0.90

Figure (21) : Throttled – Overall Response Time - Experiment 5

Algorithm	Response Time			Throughput
	Avg	Min	Max	
RR	115.19	37.63	607.62	36,513
ESCE	115.17	37.63	607.62	36,513
Throttle	114.74	37.63	607.62	36,513
PTMA	114.72	37.63	607.62	36,513

Table (10): Comparison Among all load balancing algorithms -Experiment 5

User Base	Algorithms			
	PTMA	Throttled	RR	ESCE
UB1	49.996	49.93	49.96	49.96
UB2	50.064	50.35	50.28	50.31
UB3	50.051	50	50.01	49.98
UB4	199.08	200.15	199.96	200.26
UB5	199.785	200.19	201.37	200.8
UB6	50.116	50.22	50.2	50.24
UB7	200.625	201.31	201.26	201.31

Table (11) : Comparison Among all load balancing algorithms (Average response time(ms)) - Experiment 5

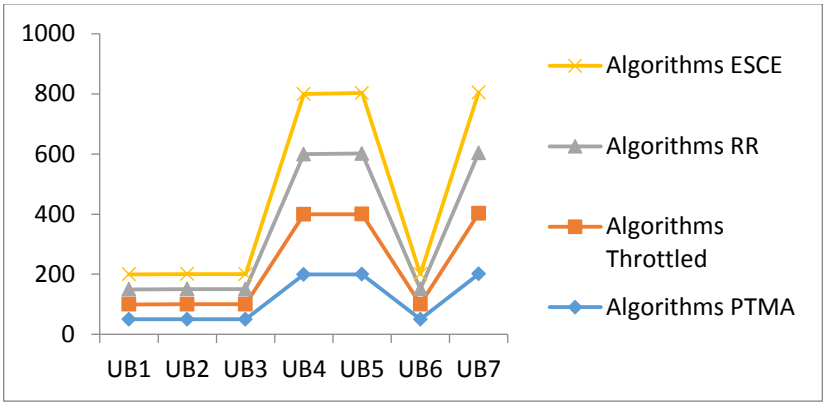


Figure (22) : Analysis showing ten user bases vs. average response time (ms) - Experiment 5

In this experiment, we found that the overall response time improved in the proposed algorithm by (0.02) ms comparing with the throttled algorithm, as well as the throughputs was (36,513) remained constant, although the overall response time is less in the proposed algorithm than the original algorithm, also there was no effect on processing time and the cost of the two algorithms.

Table(5 - 34) : Showing the summary of response time(Avg) and throughputs for all algorithms.

Experiment NO	Response Time (Avg)				Throughput
	RR	ESCE	Throttled	PTMA	
Experiment 1	71.36	71.29	71.31	71.28	42,697
Experiment 2	133.23	133.27	133.29	133.22	36,513
Experiment 3	107.84	107.83	107.87	107.81	42,697
Experiment 4	165.39	165.35	165.35	165.33	36,513
Experiment 5	115.19	115.17	114.74	114.72	36,513

Table (12): Summary of Experiments Results

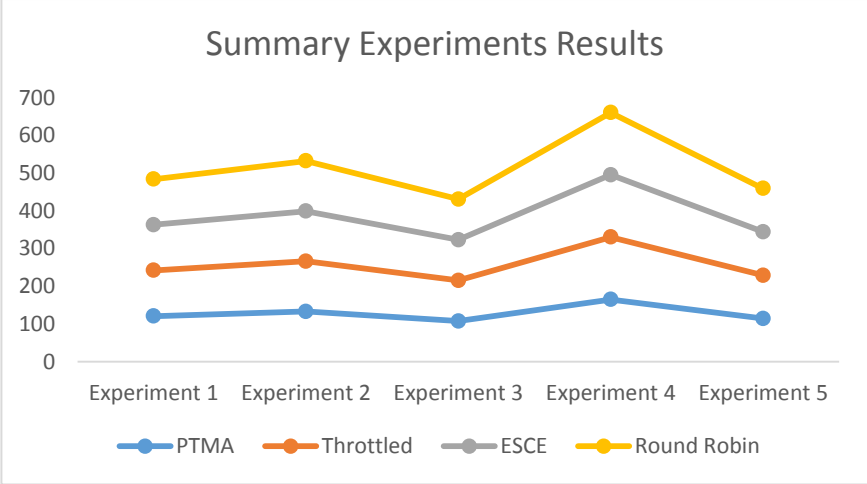


Figure (23): Summary of Experiments Results

4.3.2 Results:

There are various metrics used to evaluate the various techniques. In my work I used two metrics to measure the performance as follows:

Response Time: This is the period interval between sending a request and receiving its response. To enhance system performance, we will reduce the response time. Will get the total response time as follows:

Total response time = the users request processing delay +Network delay.[13]

Throughput: It is the number of request, which was processed at the specific period time.

4.3.2 Results Discuss:

We using Cloud Analyst simulator to conduct the experiments with the parameters configuration which explaining in the above section, we done 5 experiments. The experiment (No1, No2, No3) heterogonous environment and experiment (No4, No5)was done in homogenous environment.

In the experiments we have compared the proposed algorithm with other load balancing algorithms under two different environments, heterogeneous environment of hosts and homogenous environment. The comparison was made using two factor to evaluated the proposed algorithm: response time and throughputs. After conducting five experiment the result found

that the proposed hybrid algorithm average response time was improved all experiments, also throughputs was improved because its remained constants despite the average response time in the proposed algorithm is less others algorithm. The average response time range between (0.02 - 0.06) (ms) . This means the proposed hybrid algorithm add a significant improvement on average response time and throughput compared with other algorithms.

Overall response time was calculated by the following Equations:

Overall response time= avg. response time total/count

Here count is the number of user bases created.

Overall throughput was calculated by determined as the number of requests arrived and reposed at the duration of simulation

After comparative analysis, it is proved that proposed load balancing algorithm give better results when compared with three existing VM load balancing algorithms.

5. Future Work:

The common factor between all recently studies and researches on load balancing , focused to achieve improvement of existing algorithms by on cloud characteristics .on cloud load balance must implement at real time to serve user requests as fast as possible and maximum useful of resources utilization, so researchers and cloud providers must be take risk to go to other directions, by add in or integrate the Artificial intelligence or Machine Learning techniques into the load balance to enhance performance. This technique can be inspire in future researches.

6. Conclusion:

Load balancing is one of the critical elements for efficient operations in the cloud computing environment. In this paper, a load balancing algorithm based on throttled and Weight round robin are proposed in cloud computing. The proposed Hybrid Algorithm aims to reduce overall response time increase the throughput by optimization resources utilization. It distributes workload between various VMs taking into account the weight of each VM. The weight represented the ability of VM to process a number of tasks that is predefined early. The results is testing by cloud analyst simulator and its show that the proposed algorithm minimize the response time for requests .We also thinking that there is an improvement in throughput, although it remained constant in the results of all algorithms, we explained that because the throughput which was obtained by proposed algorithm was in a mininum response time than other algorithms. The proposed PMTA utilizes the Virtual Machines more efficiently while reducing the response time and increasing the throughputs.

References

1. Shah.M, Kariyani.A, Agrawal.D: Allocation Of Virtual Machines In Cloud Computing Using Load Balancing Algorithm, International Journal of Computer Science and Information Technology & Security (IJCSITS), ISSN: 2249-9555 Vol. 3, No.1, February 2013.
2. Sivaraj.A, Kangaiammal.A: Load Allocation of Virtual Machines based on Enhanced Throttled Load Balancing Algorithm, Turkish Journal of Computer and Mathematics Education, Vol.12 No,1841-1848,July 2021.
3. Goudarzi..Z, Faraahi.A: Effective load balancing in cloud computing, International Journal of Intelligent Information Systems,ISSN: 2328-7675 (Print); ISSN: 2328-7683 ,Vol. 3, No. 6-1, 2014,
4. Prasadhu.M ,Mehfooza.M:An Efficient Hybrid Load Balancing Algorithm forHeterogeneous Data Centers in Cloud Computing, International Journal of Advanced Trends in Computer Science and Engineering, ISSN 2278-3091,Volume 9, No.3, May - June 2020.
5. Singh.P,Baaga.P ,Gupta.S: Assorted Load Balancing Algorithms in Cloud Computing: A Survey, International Journal of Computer Applications (0975 – 8887) , Volume 143 – No.7, June 2016.
6. Devi.ch,:Uthariaraj.v.r :Load Balancing in Cloud Computing Environment Using Improved Weighted Round Robin Algorithm for Nonpreemptive Dependent Task, Scientific World Journal Volume ID 3896065, 2016.
7. Kaur.S : Load Balancing Techniques – A Review, International Journal for Research in Applied Science & Engineering Technology (IJRASET),ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor :6.887, Volume 6 Issue I, January 2018
8. James.J, Verma.B : Efficient Vm Load Balancing Algorithms For A Cloud Computing Environment, International Journal on Computer Science and Engineering (IJCSE) Vol. 4 No. 09, Sep 2012.
9. Le.H ,Tran.H : ITA: The Improve Throttled Algorithm Of Load Balancing On Cloud Computing, International Journal of Computer Networks& Communications (IJCNC) ,Vol.14, No.1, January 2022.
10. Ahmad .I, Ahmad.SH, Mirdha.S: An Enhanced Throttled Load Balancing Approach for Cloud Environment,International Research Journal of Engineering and Technology (IRJET), e-ISSN: 2395 -0056, p-ISSN: 2395-0072, Volume: 04 Issue: 06 | June -2017.
11. Shmi.B, Dr. Malhotra.D:A: Review Different Improvised Throttled Load Balancing Algorithms in Cloud Computing Environment, International Journal of Engineering Technology, Management and Applied Sciences, Volume 5, Issue 7, ISSN 2349-4476, , July 2017.
12. Phi.N,Tin.C,Thu.L,Hung.T: Proposed Load balancing Algorithm To Reduce Response Time and Processing Time on Cloud Computing , International Journal of Computer Network &&Communications(IJCNC),Vol.10,No3,May2018.
13. Jena.S, Ahmad.Z: Response Time Minimization of Different Load Balancing Algorithms in Cloud Computing Environment ,International Journal of Computer Applications (0975 – 8887) Volume 69– No.17, May 2013.
14. Kumar .M,Sharma.s: Dynamic load balancing algorithm for balancing the workload among virtual machine in cloud computing, 7th International Conference on Advances in Computing & Communications, ICACC-2017, 22-24 August 2017, Cochin, India . Published by Elsevier B.V.
15. Mohammadhossein.S, Boudec .J, Boyer.M:Interleaved Weighted Round-Robin: A Network Calculus Analysis, arXiv:2003.08372v3 [cs.NI] 24 Apr 2020.
16. Patel.D, Rajawat.A: Efficient Throttled Load Balancing Algorithm in Cloud Environment, International Journal of Modern Trends in Engineering and Research(IJMTER), Volume 02, Issue 03, e-ISSN: 2349-9745, p-ISSN: 2393-8161, March2017.
17. Manikandan.N, Pravin.A: An Efficient Improved Weighted Round Robin Load Balancing Algorithm in Cloud Computing, International Journal of Engineering & Technology, 7 (3.1) (2018) 110-117.
18. Sinha .G. Kumar Sinha.D :Enhanced Weighted Round Robin Algorithm to Balance the Load for Effective Utilization of Resource in Cloud Environment, EAI Endorsed Transactions on Cloud Systems ,Volume 6, Issue 18 e4, 05 2020 - 09 2020 .
19. A. Joshi.N: Technique for Balanced Load Balancing in Cloud Computing Environment, International Journal of Advanced Computer Science and Applications (IJACSA),Vol. 13, No. 3, 2022.
20. Pathan.A , Sneha.N: Simulation of Load Balancing Algorithms using Cloud Analyst, International Journal of Engineering Research & Technology (IJERT),NCRTS`14 Conference Proceedings(2020)
21. Khanchi.M, Tyagi.S: An Efficient Algorithm For Load Balancing in Cloud Computing, International Journal Of Engineering Sciences & Research Technology , ICTM Value: 3.00, (June, 2016).
22. S.Katiyar,Paul.R: Review on Performance Evaluation for Cloud Computing, SHODH SANGAM – A RKDF University Journal of Science and Engineering, SSN No. 2581–5806, Vol.–01, No.–02, Aug–2018.

