Optimized Load Balancing Algorithm for heterogeneous Cloud Environment

Priya Gupta[1], Er. Simarjit Virk[2]
[1]*Department of Computer Science*
Bhai Gurdas Institute of Engineering and Technology
Sangrur, India
[2]*Department of Computer Science*
Bhai Gurdas Institute of Engineering and Technology
Sangrur, India
(E-mail: angel.priyagupta@gmail.com, er.simar0126@gmail.com)

Abstract— *In cloud computing, the load balancing and scheduling is used to balance the tasks execution and to utilize the available resources in an effective manner. In order to perform load balancing and tasks scheduling in cloud computing an endless algorithm and schemes are available. In traditional load balancing machines the distribution of the jobs to the Virtual Machines (VMs) were not done in an effective manner as the VM with the highest capability has the highest chances of task allocation. Due to which the overall task migration and task completion get effected. This study develops a novel approach for load balancing in cloud computing. The task allocation to the VM is done as per the size of the task and VM collaboratively i.e. the most heaviest task is allocated to the highly capable VM and the light weight task is allocated to the small sized VM. For the simulation purpose, 50 cloudlets and 10 VMs are considered. The results are evaluated in the form of number of task migration and time taken for task completion. The evaluated results delineate that the proposed load balancing algorithm outperforms the Static round robin, weighted round robin and length based weighted round robin algorithms.*

Keywords— *Cloud Computing, Load Balancing, Resource Utilization, Round Robin Algorithm.*

I. INTRODUCTION

In a cloud computing network load balancing is achieved through reallocation of net load of the entire system among distinct nodes for achieving enhanced response time along with efficient resource utilization [1]. The load refers to network load, CPU load and the memory capacity of each server. This load is managed by using the different load balancing mechanisms [2]. The load balancing process aims at obtaining a situation in network at each point of time under which each node is neither exhausted with overload nor remains under overloaded [3-5]. The load balancing algorithms are required to assure that each and every node of the cloud is busy in processing a sort of operations [6].These load balancing techniques work as a load balancer which balances the load by distributing it to other nodes depending upon how much busy the server or node is [7]. Only the current state of the system irrespective of its previous state is taken into account by load balancing algorithms that are of dynamic in nature. Few parameters like load estimation, load comparison, stability of various systems, communication among nodes etc. are considered while choosing an appropriate algorithm [8].

II. WHY LOAD BALANCING?

The most crucial part of any network system is its load balancing capacity as it highly degrades operational capability and efficiency levels of the entire system [9].Through the use of cloud resource can be utilized in more efficient and controlled manner reducing cost to minimum level. The equal load distribution among nodes is performed. Considering the loads of 80%, 60%, 40% and 20% over four servers involved namely A, B, C, and D respectively. Through proper load distribution each server will be allocated with 50% of the total load over the entire network. The entire system can offer higher scalability to enhance the efficiency of whole distributed system by employing LB middleware [8].Despite the availability of several load balancing techniques the issue of distributing load is taken seriously due to the problem of elasticity involved in it. Different organizations offer different number of resources for the purpose of provisioning. This number of resources involved may vary company to company based over the requirement and their marketing strategies. The load balancer is responsible for optimizing the response rate for a particular request as it selects a single server which can process the request faster with greater efficiency profits among all the available servers [10].

III. BACKGROUND

Load balancing is referred as an important feature of task scheduling in clouds where tasks has allotted to the number of virtual machines. There is possibility of occurrence of a condition when Virtual Machines are overloaded, their load must be transferred to the other machines who are under loaded in order to accomplish the proper utilization of resources available with least computation time. This task has

done through several existing load balancing algorithms such as Round Robin, Size based round robin and improved round robin. These algorithms provide balancing of loads to the other virtual machines in an ineffective manner.

The main problem in the existing system was that all the algorithms perform load balancing with high computation time. Thus, their computation cost can be reduced further with the introduction of a novel approach. The FCFS algorithm has done load balancing in the form of queue regardless of the load on the virtual machine. Thus, it does not consider the priority, size of the tasks and resource capability which leads to the higher response time. Moreover, low virtual machine has been allotted with a maximum load task which resultant into high load on the corresponding virtual machine. Now consider the size based round robin algorithm where the larger numbers of tasks are allotted to the weightiest Virtual machine and then others machines with lowest loads sets as free. Furthermore, the proper utilization of resources is not achieved. Lastly, consider improved round robin machine where the problem exists in selecting best virtual machine for each packet and consequently each number of time the best machine i.e. similar virtual machine has chosen for each packet. In addition to this, load lay on a single machine always and reduces the optimality of the system. Following is the traditional load balancing algorithm.

Traditional Load Balancing Algorithm

1. Identify the number of executing/pending tasks in each VM and arrange it in increasing order on a Queue.
   (a) Set $numTaskInQueue = Number\ of\ Executing/Waiting\ Tasks$ in each VM and arrange it in increasing order
2. If the number of tasks in the first item of the queue is greater than or equal to "1", then terminate the Load Balancing logic execution else proceed to the 3rd step.
   (a) $If\ (numTaskInQueue.first() \geq 1)\ then$

Return;

3. If the number of tasks in the last item of the queue is less than or equal to "1", then terminate the Load Balancing logic execution else proceed to the 4th step.
   (a) $If\ (numTaskInQueue.last() \leq 1)\ then$
   Return;
4. Identify the Pending Execution Time in each of the VMs by adding the Pending Execution length from executing, waiting & paused list and then divided the value by the processing capacity of the VM.
   (a)$Set\ pendingJobsTotLen = JobsRemLenInExecList + JobsRemLenInWaitList + JobsRemLenInPauseList$
   (b)S$et\ pendingExecutionTime = pendingJobsTotLenth/CVm$
5. Arrange the VMs based on the least pending time to the highest pending time and group it, in case two VMs fall in the same pending time.
   (a) Sort the $VMMap$ by the Pending Execution time of each VM
6. Remove a task from the higher pending time VM, which contains more than one task and assign this task to the lower pending time VM, which has no task to process.
   (a)While (true)

$Set\ OverLoadedVM = VMMap.get(VMMap.size())$

$Set\ LowLoadedVM = VMMap.get(0)$

$Varlowerposition = 1;$

$Varupperposition = 1;$

(b)While (true)

$If\ (OverLoadedVM.taskSize() > 1\ \&\&LowLoadedVM.taskSize() < 1)$

Break;

$Else\ if\ (OverLoadedVM.taskSize() > 1)$

$LowLoadedVM = VMMap.get(lowerposition)$

Lowerposition++

$Else\ if\ (LowLoadedVM.taskSize() < 1)$

$OverLoadedVM = VMMap.get(VMMap.size() - upperposition)$

Upperposition++

Else

$BreakThe\ OuterWhile\ Loop$

(c) EndWhile

$Set\ migratableTask = OverLoadedVM.getMigratableTask()$

$LowLoadedVM.assign(migratableTask)$

Break

(d) End While

7. Re execute from the step 1.
8. Then the steps 2 and 3 will decide the load balancing further.
9. This load balancing will be called after every task completion irrespective of any VMs.

In above defined load balancing algorithm it has been seen that the task allocation is done on the basis of the load handling capacity of the machine i.e. the VM with high capability has the highest chances for large number of task allocation and other VMs with lowest capability will remain empty or free. Thus in this manner this strategy of load balancing leads to the increment in the overheads and cost as well as it leads to the less efficiency in task completion by the VMs as a single VM has to handle a large number of tasks due to which its caliber to perform processing of tasks also reduces.

IV. PROPOSED WORK

From the literature survey, it has computed that conventional load balancing algorithms have the capability in balancing load among different virtual machines but these techniques are not capable enough in reducing the computation cost with sharing of loads. Thus, considering this fact, a new technique has to be proposed which can divide up the load among other virtual machines in an effective and efficient manner and capable of producing optimal results.

The existing techniques have been facing the issue of finding a virtual machine for their task, so a new solution has proposed. In this new proposed technique two solutions have been carried out: according to the virtual machine load will be distributed i.e. higher load will not be assigned to the lowest machine and vice versa. Moreover, equal load has been shared among different virtual machines i.e. a problem of starvation for any VM will not be occurred.

Proposed Load Balancing Algorithm

1. Identify the number of under processing tasks in each VM and arrange it in order.
   (a) $Set\ CountTask = Number\ of\ processing\ Tasks\ in\ each\ VM$

2. If the number of tasks in the first and last item of the queue is greater than or equal to "1", then terminate the Load Balancing logic execution else proceed to the 3rd step.
   (a)$If\ (CountTask.first() \geq 1)\ or\ (CountTask.last() \geq 1)\ then$
   Return;
3. Calculate the pending time available in each of the VMs , later also check the length of the cloudlets available to be executed, and then divided the value by the processing capacity of the VM.
   $(a) Set\ pendingExecutionTime = ActualVMexecutiontime - TimeComcsumed\ for\ CountTasks;$
   (b)$Set\ AvailableTaskToPerform = sum(Time\ of\ each\ Task)$
   (c) $Set\ RequiredTimeSlot = AvailableTaskToPerform / pendingExecutionTime;$
4. Arrange the VMs based on the maximum time availability to the minimum time availability and group it, in case two VMs fall in the same pending time. Sort the VMMap by the same order as they are in original VMmap.
5. Assign the cloudlet of high execution time to the higher availability time VM, and assign the low execution task to the lower available time VM, which has no task to process, or less available time.
   (a)While (true)
   $Set\ MaxAvailTimeVM = VMMap.get(VMMap.size())$
   $Set\ MinAvailTimeVM = VMMap.get(0)$
   $Varlowerposition = 1;$
   $Varupperposition = 1;$
   (b)While (true)
   $If\ (MaxAvailTimeVM.taskSize() > Availtime\ \&\&\ MinAvailTimeVM.taskSize() < Availtime)$
   Break;
   $Else\ if\ (MaxAvailTimeVM.taskSize() > Availtime)$
   $MinAvailTimeVM = VMMap.get(lowerposition)$
   Lowerposition++
   $Else\ if\ (MinAvailTimeVM.taskSize() < Availtime)$
   $MaxAvailTimeVM = VMMap.get(VMMap.size() - upperposition)$
   Upperposition++
   Else
   $BreakThe\ OuterWhile\ Loop$
   (c) EndWhile
   $Set\ migratableTask = MaxAvailTimeVM.getMigratableTask()$
   $MinAvailTimeVM.assign(migratableTask)$
   Break
   (d) End While
6. $FinalAssignArray = [MaxAvailTimeVM.assign\ MinAvailTimeVM.assign]$
7. This load balancing will be called after every task completion irrespective of any VMs

The following figure depicts the framework for the proposed load balancing algorithm in cloud computing. The process is started with the user. The user interacts with the interface and then interface sets the job queue according to the incoming tasks. After this the tasks are divided into two queues one for dependency task queue and another for independent tasks. After this scheduler coordinates with the resource manager for resource allocation to the tasks. The resource manager looks for the available resources and then the proposed load balancing scheme is applied to manage resource allocation efficiently. On the basis of the proposed load balancing algorithm the heavy task is assigned to the heavy virtual machine and the small task is assigned to the small virtual machines. In this manner the load balancing is done and the tasks execution runs smoothly.

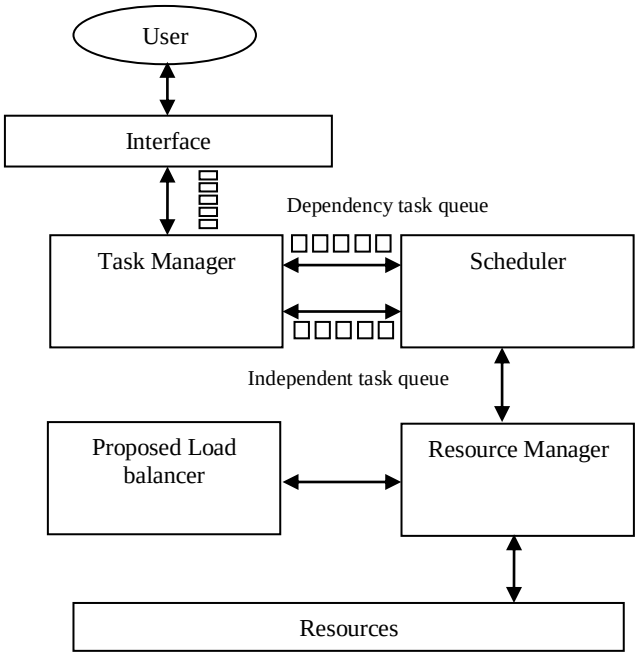

Figure 1 Proposed Load Balancing Framework

V. RESULTS ANALYSIS

This work implements the load balancing algorithm for 50 cloud lets. Total 10 virtual machines are considered for the purpose of implementation and to share the load of cloud lets. The load balancing is done on the basis of the size of the virtual machine and jobs or cloud lets. The heavy job is assigned to the most suitable and highly capable virtual machine. Java programming platform is used for the purpose of simulation.

For the purpose of easy access, a user interface is created. In proposed work, a comparison analysis is done among static round robin load balancing algorithm, weighted round robin algorithm; Length based weighted round robin algorithm and proposed load balancing algorithm. The performance of proposed work is evaluated in the terms of space shared and time shared. In space shared the jobs are executed in a sequence order. As per the space shared mechanism the CPU executes the single job at the given time of interval and for the timing the rest of the allotted jobs have to wait for their turn of execution. A queue is maintained for the jobs that are waiting for their turn for execution. Thus in this manner the task

migration becomes easier as the task can directly switch from queue to available virtual machine.

The graph in figure 2 shows the comparison of this mechanism in the terms of number of task migrations by using the space shared mechanism. The number of task migration in static round robin technique is higher in comparison to the other techniques. Whereas the number of task migration in proposed mechanism lower and effective.

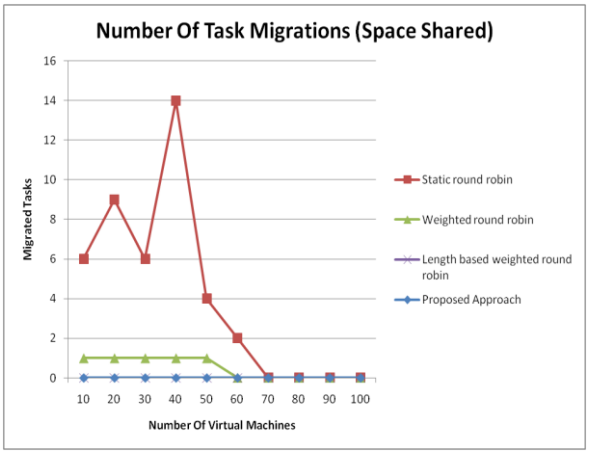

Figure 2 Comparison Analysis of Task Migration (Space shared)

The graph in figure 3 depicts the comparison of task migration with respect to the time shared mechanism for static round robin, weighted round robin, and length based weighted round robin and proposed work. Initially the number of task migration in all of the considered mechanism is lower but with the increment in the number of virtual machines, the number of task migration for static round robin and weighted round robin also gets higher. In this case the number of task migration is evaluated to be higher for weighted round robin algorithm but the proposed approach and length based round robin has lower task migration rate.

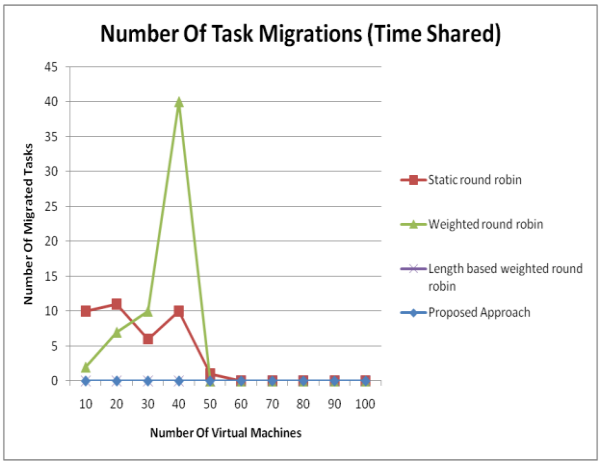

Figure 3 Comparison Analysis of Task Migration (Time shared)

The graph in figure 4 and 5 depicts the comparison of traditional round robin based load balancing mechanisms and proposed load balancing algorithm. The comparison analysis is done on the basis of the rate of task overall completion time. The comparison on figure 5.1 is analyzed on the basis of the space shared task migration technique and in figure 5the comparison is done on the basis of the time shared task migration mechanism. In both cases, on the basis of the observations, it is concluded that the overall task completion time for proposed work is lower in contrast to the rest of the techniques.

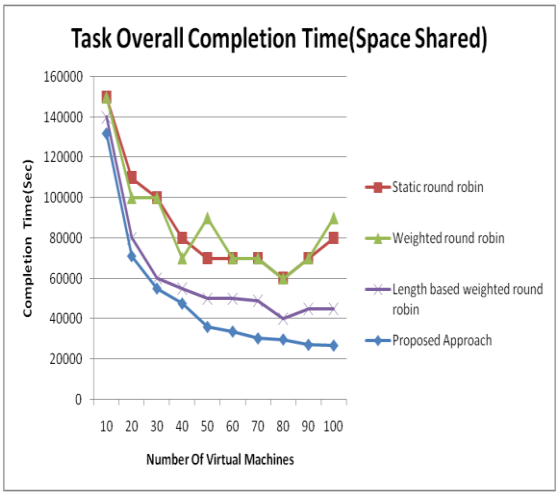

Figure 4 Overall Time taken for task completion (Space Shared)

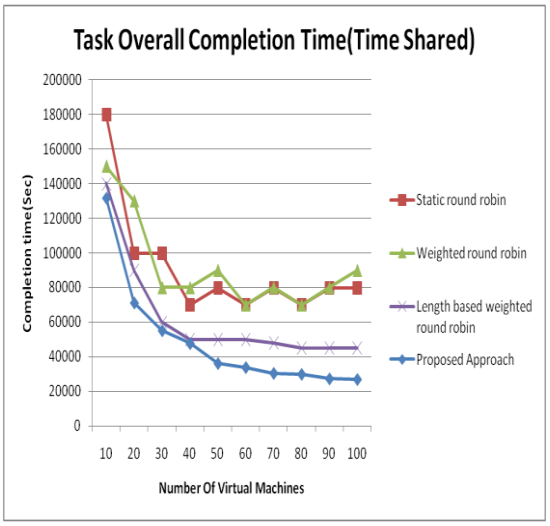

Figure 5 Overall Time taken for task completion (Space Shared)

The table 1 calibrates the facts and figures that are observed from the comparison graph of the figure 5.9. The table depicts the number of migrated tasks by the virtual machines while applying the various load balancing algorithms. The static round robin technique is found to have higher number of task migrations by different virtual machines. Whereas the proposed work and length based weighted round robin algorithm the number of task migration is observed to be 0 during

various virtual achiness. Similarly the table 2 calibrates the number task migrating performed during various load balancing techniques under time shared mechanism. In this case the proposed work is found to be effective in all cases i.e. for different virtual machines.

Table 1 Number of Task Migration (Space Shared)

Number Of Virtual Machines	Proposed Approach	Static round robin	Weighted round robin	Length based weighted round robin
10	0	6	1	0
20	0	9	1	0
30	0	6	1	0
40	0	14	1	0
50	0	4	1	0
60	0	2	0	0
70	0	0	0	0
80	0	0	0	0
90	0	0	0	0
100	0	0	0	0

Table 2 Number of Task Migration (Time Shared)

Number Of Virtual Machines	Proposed Approach	Static round robin	Weighted round robin	Length based weighted round robin
10	0	10	2	0
20	0	11	7	0
30	0	6	10	0
40	0	10	40	0
50	0	1	0	0
60	0	0	0	0
70	0	0	0	0
80	0	0	0	0
90	0	0	0	0
100	0	0	0	0

Table 3 and 4 depicts the facts and figures that are observed after analyzing the overall time taken to complete the task by various load balancing algorithms. The overall completion time should be low for an ideal load balancing algorithm. Thus the tables conclude that the time taken by proposed work in both cases i.e. time shared and space shared is lower in comparison to the rest of the techniques.

Table 3 Task Overall Completion time (Space Shared)

Number Of Virtual Machines	Proposed Approach	Static round robin	Weighted round robin	Length based weighted round robin
10	131761.61	150000	150000	140000
20	71196.78	110000	100000	80000
30	55192.43	100000	100000	60000
40	47745.832	80000	70000	55000
50	36215.22	70000	90000	50000
60	33919.56	70000	70000	50000
70	30544.49	70000	70000	49000
80	29995.77	60000	60000	40000
90	27369.452	70000	70000	45000
100	27000	80000	90000	45000

Table 4 Task Overall Completion time (Time Shared)

Number Of Virtual Machines	Proposed Approach	Static round robin	Weighted round robin	Length based weighted round robin
10	131761.61	180000	150000	140000
20	71196.78	100000	130000	90000
30	55192.43	100000	80000	60000
40	47745.832	70000	80000	50000
50	36215.22	80000	90000	50000
60	33919.56	70000	70000	50000
70	30544.49	80000	80000	48000
80	29995.77	70000	70000	45000
90	27369.452	80000	80000	45000
100	27000	80000	90000	45000

VI. CONCLUSION

The cloud computing comes to the existence to secure the storage spaces and various other computing resources. It is a technology that serves as a blessing to present generation of computerization. In cloud computing the user is facilitated with software, platform, and infrastructure as a service via a service provider. The cloud computing has become most trending field for research work due its popularity and increased use. The cloud is made up of various components or resources that are unalike to each other. The cost incurred to execute the operations or jobs in a cloud also varies and indirectly it relies on utilization of the resources. Therefore to maintain the cost and utilization of resources in the cloud, the load balancing and scheduling is mandatory.

To sum up, this study develops a load balancing scheme in which the load of job is distributed among the virtual machines on the basis of the capability of the machines. For the purpose of implementation, JAVA programming platform is used. For the purpose of implementation, total 50 cloudlets and 10 virtual machines are considered. The results are evaluated in the form of task completion time and task migration rate. Th comparison analysis of proposed work is done with static round robin, weighted round robin and length based weighted round robin algorithm for load balancing. On the basis of the results, it is observed that the proposed work

has less migrated task and high task completion time in contrast to traditional load balancing algorithms.

In future more enhancements in the present work can be introduced by using the swarm based intelligent optimization techniques for scheduling the resources and balancing and allocating the tasks to the VMs. In this way the performance of the cloud system would also get more optimized and efficient.

REFERENCES

[1]. D. Chitra Devi et al, "Load Balancing in Cloud Computing Environment Using Improved Weighted Round Robin Algorithm for Nonpreemptive Dependent Tasks", Hindawi, Vol. 2016, Pp. 1-14, 2016

[2]. Einollah Jafarnejad Ghomi et al, "Load-balancing algorithms in cloud computing: A survey", Journal of Network and Computer Applications, Vol. 88, Pp. 50-71, June 2017

[3]. Lipika Datta, "Efficient Round Robin Scheduling Algorithm with Dynamic Time Slice", I.J. Education and Management Engineering, Vol. 2, Pp. 10-19, 2015

[4]. Saraswathi Seemakuthi et al, "A Review on Various Scheduling Algorithms", International Journal of Scientific & Engineering Research, Vol. 6, No. 12, Pp. 769-779, December 2015

[5]. Shalini Joshi et al, "Load balancing in cloud computing: Challenges & issues", Contemporary Computing and Informatics (IC3I), 2016 2nd International Conference on, May 2017

[6]. Ojasvee Kaneria et al, "Analysis and improvement of load balancing in Cloud Computing", ICT in Business Industry & Government (ICTBIG), International Conference on, April 2017

[7]. Monjur Ahmed et al, "Cloud Computing And Security Issues in The Cloud", IJNSA, Vol 6, Issue 1, Pp 25-36, 2014

[8]. Michael armbrust,"A view on Cloud computing", communication of the ACM, vol 53(4), 2009

[9]. Ling Qian,"Cloud computing-An overview", springer, Pp 626-631, 2009

[10]. Matthias Sommer et al, "Predictive Load Balancing in Cloud Computing Environments Based on Ensemble Forecasting", Autonomic Computing (ICAC), 2016 IEEE International Conference on, September 2016

[11]. Aarti Vig et al, "An Efficient Distributed Approach for Load Balancing in Cloud Computing", Computational Intelligence and Communication Networks (CICN), 2015 International Conference on, August 2016

[12]. Sidra Aslam et al, "Load balancing algorithms in cloud computing: A survey of modern techniques", Software Engineering Conference (NSEC), 2015 National, February 2016

[13]. Reena Panwar et al, "Load balancing in cloud computing using dynamic load management algorithm", Green Computing and Internet of Things (ICGCIoT), 2015 International Conference on, January 2016

[14]. Agraj Sharma et al, "Response time based load balancing in cloud computing", Control, Instrumentation, Communication and Computational Technologies (ICCICCT), 2014 International Conference on, December 2014

[15]. Velagapudi Sreenivas et al, "Load balancing techniques: Major challenge in Cloud Computing - a systematic review", Electronics and Communication Systems (ICECS), 2014 International Conference on, September 2014

[16]. Vishnu Kumar Dhakad et al, "Performance Analysis Of Round Robin Scheduling Using Adaptive Approach Based On Smart Time Slice And Comparison With SRR", International Journal of Advances in Engineering & Technology, Vol. 3, No. 2, Pp. 333-339, May 2012.

[17]. Mohammad Shoaib et al, "A Comparative Review of CPU Scheduling Algorithms", Proceedings of National Conference on Recent Trends in Parallel Computing, November 2014

[18]. Maniyar Bhumi J et al, "Review On Round Robin Algorithm For Task Scheduling In Cloud Computing", International Journal of Emerging Technologies and Innovative Research, Vol. 2, No. 3, Pp. 788-793, March 2015

[19]. D. Saranya et al, "Load Balancing Algorithms in Cloud Computing: A Review", International Journal of Advanced Research in Computer Science and Software Engineering, vol. 5, No. 7, Pp. 1107-1111, July 2015

[20]. Dharmesh Kashyap et al, "A Survey of Various Load Balancing Algorithms in Cloud Computing", International Journal Of Scientific & Technology Research, Vol. 3, No. 11, November 2014

[21]. Mayanka Katyal et al, "A Comparative Study of Load Balancing Algorithms in Cloud Computing Environment", International Journal of Distributed and Cloud Computing Vol. 1, No. 2, Pp. 5-14, December 2013

[22]. Sushil Kumar et al, "Various Dynamic Load Balancing Algorithms in Cloud Environment: A Survey" , International Journal of Computer Applications, Vol. 129, No. 6, Pp. 14-19, November 2015

[23]. Komal Purba et al, "A Review on Load Balancing Algorithm in Cloud Computing", SSRG International Journal of Computer Science and Engineering, Vol. 1, No. 10, Pp. 19-23, December 2014

[24]. Alireza Sadeghi Milani et al, "Load balancing mechanisms and techniques in the cloud environments: Systematic literature review and future trends", Journal of Network and Computer Applications, Vol. 71, Pp. 86-98, August 2016

[25]. P.P. Geethu Gopinath et al, "An In-depth Analysis and Study of Load Balancing Techniques in the Cloud Computing Environment", Procedia Computer Science, Vol. 50, Pp. 427-432, 2015