A Comparison of Four Popular Heuristics for Load Balancing of Virtual Machines in Cloud Computing

Subasish Mohapatra
Department Of CSE
NIT, ROURKELA

K.Smruti Rekha
Department Of CSE
ITER, SOA UNIVERSITY

Subhadarshini Mohanty
Department Of CSE
ITER, SOA UNIVERSITY

ABSTRACT

Cloud computing is an internet based computing. This computing paradigm has enhanced the use of network where the capability of one node can be utilized by other node. Cloud provides services on demand to distributive resources such as database, servers, software, infrastructure etc. in pay as you go basis. Load balancing is one of the vexing issues in distributed environment. Resources of service provider need to balance the load of client request. Different load balancing algorithms have been proposed in order to manage the resources of service provider efficiently and effectively. This paper presents a comparison of various policies utilized for load balancing.

Keywords: Cloud computing, virtual machines, cloud service provider, load balancing, cloud analyst.

1. INTRODUCTION

Cloud computing is one of the internet based service provider which allows users to access services on demand [1]. It provides pool of shared resources of information, software, databases and other devices according to the client request on "pay as you go" basis [2]. Cloud computing architectures are inherently parallel, distributed and serve the needs of multiple clients in different scenarios. This distributed architecture deploys resources distributively to deliver services efficiently to users in different geographical channels [3]. Clients in a distributed environment generate request randomly in any processor. So the major drawback of this randomness is associated with task assignment. The unequal task assignment to the processor creates imbalance i. e, some of the processors are overloaded and some of them are underloaded [4]. The objective of load balancing is to transfer the load from overloaded process to underloaded process transparently. The major issues associated with load balancing are flexibility and reliability of services [5]. Cloud computing implements virtualization technique in which a single system can be virtualized into number of virtual systems [6]. Load balancing decides which client will use the virtual machine and which requesting machines will be put on hold. Load balancing of the entire system can be handled dynamically by using virtualization technology where it becomes possible to remap Virtual Machines (VMs) and physical resources according to the change in load. Due to these advantages, virtualization technology is being comprehensively implemented in Cloud computing. A Virtual Machine (VM) is a software implementation of a computing environment in which an operating system (OS) or program can be installed and run. The Virtual Machine typically changes a physical computing environment and requests for CPU, memory, hard disk, network and other hardware resources are managed by a virtualization layer which translates these requests to the underlying physical hardware. VMs are created within a virtualization layer, such as a hypervisor or a virtualization platform that runs on top of a client or server operating system. This operating system is known as the host OS. The virtualization layer can be used to create many individual, isolated VM environments, where multiple requests or tasks can execute in multiple machines [7].

1.1 Load balancing algorithms

Load balancing [8] algorithm directly influences the effect of balancing the server workloads. Its main task is to decide how to choose the next server node and transfer a new connection request to it. Current main load balancing algorithm is divided into static algorithm and dynamic algorithm [9]. The static algorithm is easily carried into execution and takes less time, which doesn't refer to the states of the load nodes, but it can be only used in certain specific conditions. The common static algorithms are Round-Robin Scheduling Algorithm, Weighted Round-Robin Scheduling Algorithm, and Least-Connection Scheduling Algorithm etc. Of all, Round -Robin Scheduling Algorithm is the simplest one which could be most easily be carried out. However, it is only applicable to the circumstances in which all the nodes in cluster have the same processing ability. The dynamic algorithm like first come first serve is self-adaptive algorithm, which is better than static algorithm, and suitable for a great deal of requests which procreate different workloads, which would be unable to be forecasted [10]. Self-adaptive load balancing system mainly includes two processes: monitoring the load states of servers and assigning the request to the servers. The state supervision, which depends on the load information of each node in the cluster monitored and collected periodically by the front-end scheduler, raises the effect of load balance by monitoring load variety. At the same time, assigning the load carries on synchronous operation according to the load information from all nodes, that is, redistributing the load that needs to be done.

According to the analysis above, the ideal load balancing algorithm should achieve the following targets:

- Leave the collections, computing of load node information for each node; prevent the front-end scheduler from being system bottleneck.
- Reduce the complications of load balancing algorithm as far as possible.

2. RELATED WORK

Cloud computing is recent emerging technology in IT industry leading towards the researches advances in many domains. Jiyni et.al, (2010) have proposed a resource allocation mechanism with preemptable task execution which increases the utilization of clouds. They have proposed an adaptive resource allocation algorithm for cloud system with preemptable tasks but their approach does not pertain to cost optimization and time optimization [11]. M. Randles etal have proposed comparison of static and dynamic load balancing algorithm for cloud computing. [12]. Ram Prasad Padhy, P Goutam Prasad Rao discussed on basic concepts of Cloud Computing and Load balancing and studied some existing load balancing algorithms, which can be applied to clouds [5]. In addition to that, the closed-form solutions for minimum measurement and reporting time for single level tree networks with different load balancing strategies were also studied [13]. The performance of these strategies with respect to the timing and the effect of link and measurement speed were studied. The paper described the features of a simulator to compare the performance of three dynamic load balancing algorithms. Cloud Analyst: A Cloud Sim-based Visual Modeller for Analysing Cloud Computing Environments and Applications [14] Bhathiya Wickrema Singh all present how Cloud Analyst can be used to model and evaluate a real world problem through a case study of a social networking application deployed on the cloud. We have illustrated how the simulator can be used to effectively identify overall usage patterns and how such usage patterns affect data centres hosting the application [15].

3. HEURISTICS OF LOAD BALANCING

3.1 Round Robin

It is one of the simplest scheduling techniques that utilize the principle of time slices. Here the time is divided into multiple slices and each node is given a particular time slice or time interval i.e. it utilizes the principle of time scheduling [16]. Each node is given a quantum and in this quantum the node will perform its operations. The resources of the service provider are provided to the requesting client on the basis of this time slice. The following figure shows how round robin works. The following figure shows that each user request is served by every processor within given time quantum. After the time slice is over, the next queued user request will come for execution. If the user request completes within time quantum then user should not wait otherwise user have to wait for its next slot.

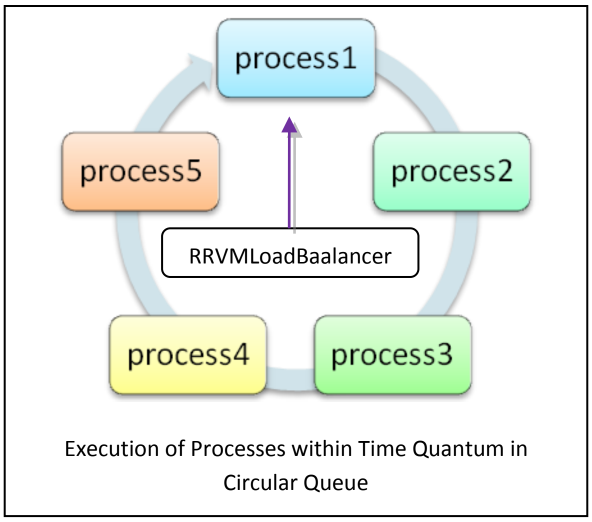
process1
process2
process3
process4
process5
RRVMLoadBaalancer
Execution of Processes within Time Quantum in Circular Queue

Fig 1: Round Robin processing

ROUND ROBIN ALGORITHM

1. Round RobinVmload Balancer maintains an index of VMs and state of the Vms (busy/available). At start all vm's have zero allocation.
2. a. The datacentercontroller receives the user requests/cloudlets.
   b. It stores the arrival time & burst time of the user requests.
   c. The requests are allocated to Vms on the basis of their states known from the VM queue.
   d. The roundrobinvmloadbalancer will allocate the time quantum for user request execution.
3. a. The roundrobinvmloadbalancer will calculate the turn- around time of each process.
   b. It also calculates the response time and average waiting time of user requests.
   c. It decides the scheduling order.
4. After the execution of cloudlets, the VMs are de- allocated by the RoundRobinVmLoadBalancer.
5. The datacentercontroller checks for new /pending/waiting requests in queue.
6. Continue from step-2.

3.2 Throttled

In this algorithm the client first requests the load balancer to find a suitable Virtual Machine to perform the required operation. It is shown in fig.2.The process first starts by maintaing a list of all the VMs each row is individually indexed to speed up the lookup process. If a match is found on the basis of size and availability of the machine, then the load balancer accepts the request of the client and allocates that VM to the client. If, however there is no VM available that matches the criteria then the load balancer returns -1 and the request is queued. The following figure shows how it works [16].

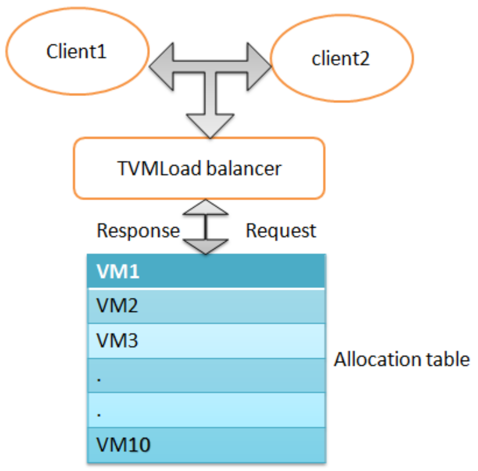

Fig 2: Throttled scheduling process

THROTTLED ALGORITHM

1. ThrottledVmLoadBalancer maintains an index table of VMs and the state of the VM (BUSY/AVAILABLE). At the start all VM's are available.
2. DataCenterController receives a new request.
3. DataCenterController queries the ThrottledVmLoadBalancer for the next allocation.
4. ThrottledVmLoadBalancer parses the allocation table from top until the first available VM is found or the table is parsed completely.
   If found:
   i) The ThrottledVmLoadBalancer returns the VM id to the DataCenterController.
   ii) The DataCenterController sends the request to the VM identified by that id.
   iii) DataCenterController notifies the ThrottledVmLoadBalancer of the new allocation.
   iv) ThrottledVmLoadBalancer updates the allocation table accordingly.

   If not found:

   i) The ThrottledVmLoadBalancer returns -1.
   ii) The DataCenterController queues the request.
5. When the VM finishes processing the request, and the DataCenterController receives the response cloudlet, it notifies the ThrottledVmLoadBalancer of the VM de-allocation.
6. The DataCenerController checks if there are any waiting requests in the queue. If there are, it continues from step 3.
7. Continue from step 2.

3.7 Execution Load

It is spread spectrum technique in which the load balancer spread the load of the job in hand into multiple virtual machines. The load balancer maintains a queue of the jobs that need to use and are currently using the services of the virtual machine. The balancer then continuously scans this queue and the list of virtual machines. If there is a VM available that can handle request of the node/client, the VM is allocated to that request [17]. If however there is a VM that is free and there is another VM that needs to be freed of the load, then the balancer distributes some of the tasks of that VM to the free one so as to reduce the overhead of the former VM. The jobs are submitted to the VM manager, the load also maintains a list of the jobs, their size and the resources requested. The balancer selects the job that matches the criteria for execution at the present time. Though there algorithm offers better results as shown in further section, it however requires a lot of computational overhead. The following figure shows how ESCEL works.

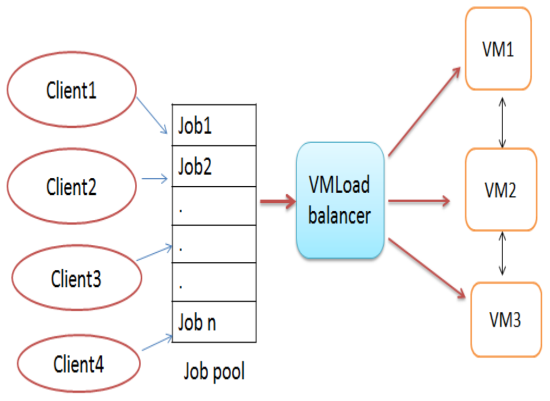

Fig 3: ESCEL scheduling process

EQUALLY SPREAD CURRENT EXECUTION LOAD

1. Find the next available VM
2. Check for all current allocation count is less than max length of VM list allocate the VM
3. If available VM is not allocated create a new one
4. Count the active load on each VM
5. Return the id of those VM which is having least load.
6. The VMLoadBalancer will allocate the request to one of the VM.
7. If a VM is overloaded then the VMLoadBalancer will distribute some of its work to the VM having least work so that every VM is equally loaded.
8. The datacentercontroller receives the response to the request sent and then allocate the waiting requests from the job pool/queue to the available VM & so on.
9. Continue from step-2.

3.8 First Come First Serve

FCFS (First Come First Served), used in parallel task processing, is the simplest task ordering strategy. It chooses and processes them according to the right order of jobs getting into system [18]. With this scheme the user request which comes first to the datacentercontroller is allocated the virtual machine for execution first. The implementation of FCFS policy is easily managed with FIFO queue. The datacentercontroller searches for virtual machine which is in idle state or underloaded. Then the 1st request from the queue is removed and passed to one of the VM through the VMLoadBalancer. The following figure shows how FCFS works.

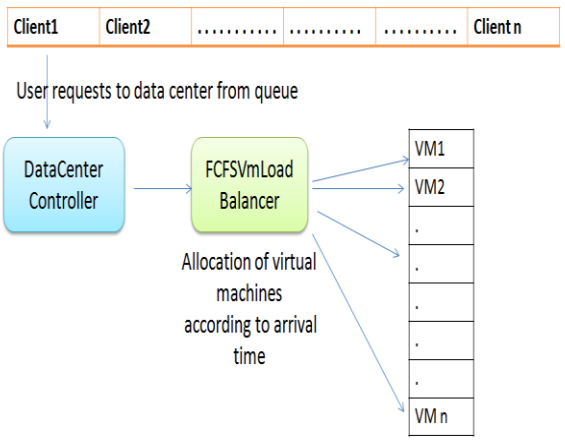

Fig 4: FCFS scheduling process

<u>FIRST COME FIRST SERVE ALGORITHM</u>

1. FCFS VmloadBalancer maintains an index table of virtual machines & number of requests currently allocated to the VM. At start all have zero allocation.
2. a) The vmloadbalancer allocates the cloudlets/user requests to the available VMs on the basis of requests sent by the datacentercontroller.
   b) The datacentercontroller stores the user requests in a queue on the basis of their arrival time.
   c) The first request according to the arrival time is allocated to the VM which is under loaded or free by FCFSVmloadBalancer.
3. The FCFSVmLoadBalancer will execute the cloudlets and calculate the turnaround time, avg. waiting time and response time. After that it will display the result.
4. The datacentercontroller receives the response to the request sent and then allocate the waiting requests from the job pool/queue to the available VM & so on.
5. Continue from step-2.

4. CLOUD ANALYST

Cloud Analyst [14] [16] [17] is a GUI based tool that is developed on CloudSim architecture. CloudSim is a toolkit that allows doing modelling, simulation and other experimentation. The main problem with CloudSim is that all the work need to be done programmatically. It allows the user to do repeated simulations with slight change in parameters very easily and quickly. The cloud analyst allows setting location of users that are generating the application and also the location of the data centers. In this various configuration parameters can be set like number of users, number of request generated per user per hour , number of virtual machines, number of processors, amount of storage, network bandwidth and other necessary parameters. Based on the parameters the tool computes the simulation result and shows them in graphical form. The result includes response time, processing time, cost etc .By performing various simulations operation the cloud provider can determine the best way to allocate resources, based on request which data center to be selected and can optimize cost for providing services.

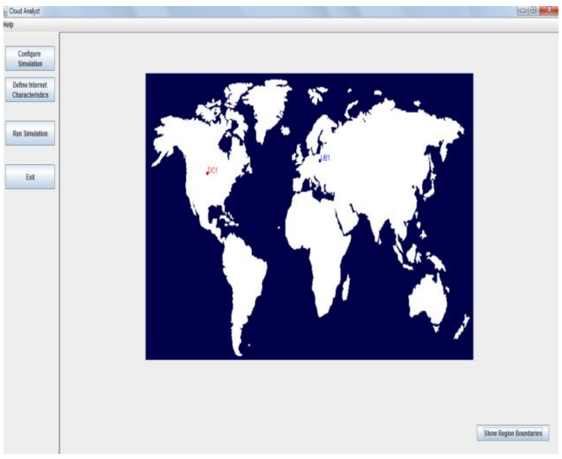

Fig 5: GUI interface

4.1 Simulation Parameters:

4.1.1 Region

In the CloudAnalyst the world is divided in to 6 'Regions' that coincide with the 6 main continents in the World. The other main entities such as User Bases and Data Centers belong to one of these regions. This geographical grouping is used to maintain a level of realistic simplicity for the large scaled simulation being attempted in the CloudAnalyst.

4.1.2 Users

A User Base models a group of users that is considered as a single unit in the simulation and its main responsibility is to generate traffic for the simulation. A single User Base may represent thousands of users but is configured as a single unit and the traffic generated in simultaneous bursts representative of the size of the user base. The modeller may choose to use a User Base to represent a single user, but ideally a User Base should be used to represent a larger number of users for the efficiency of simulation.

4.1.3 DataCenterController

The Data Center Controller is probably the most important entity in the CloudAnalyst. A single Data Center Controller is mapped to a single cloudsim. DataCenter object and manages the data center management activities such as VM creation and destruction and does the routing of user requests received from User Bases via the Internet to the VMs. It can also be viewed as the façade used by CloudAnalyst to access the heart of CloudSim toolkit functionality.

4.1.4 InternetCharacteristics

In this component various internet characteristics are modelled simulation, which includes the amount of latency and bandwidth need to be assigned between regions, the amount of traffic, and current performance level information for the data centers.

4.1.5 VmLoadBalancer

The responsibility of this component is to allocate the load on various data centers according to the request generated by users. One of the f our given policies can be selected. The given policies are round robin algorithm, equally spread current execution load, throttled and first come first serve.

4.1.6 CloudAppServiceBroker

The responsibility of this component is to model the service brokers that handle traffic routing between user bases and data centers. The service broker can use one of the routing policies from the given three policies which are closest data center, optimize response time and reconfigure dynamically with load. The closest data center routes the traffic to the closest data center in terms of network latency from the source user base. The reconfigure dynamically with load routing policy works in the sense that whenever the performance of particular data center degrades below a given threshold value then the load of that data center is equally distributed among other data centers.

In order to analyze various load balancing policies configuration of the various component of the cloud analyst tool need to be done. We have set the parameters for the user base configuration, application deployment configuration, and data center configuration as shown in figure 6. As shown in figure the location of user bases has been defined in six different regions of the world. We have taken four data centers to handle the request of these users. On DC1 there are 25 VMs allocated, 50 VMs are allocated to DC2, 75 VMs are allocated to DC3 and 100 VMs are allocated to DC4 respectively. Here we have taken 6 user bases.

Table1: Setting user bases and data center

DATA CENTER	VMS	IMAGE SIZE	MEMORY	BW
DC1	25	10000	512	1000
DC2	50	10000	512	1000
DC3	75	10000	512	1000
DC4	100	10000	512	1000

USERS	REGION	REQUET 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	0	60	100	3	9	1000	100
UB2	1	60	100	3	9	2000	100
UB3	2	60	100	3	9	3000	100
UB4	3	60	100	3	9	4000	100
UB5	4	60	100	3	9	5000	100
UB6	5	60	100	3	9	6000	100

5. RESULT AND RESPONSE TIME

After performing the simulation the result computed by cloud analyst is as shown in the following figures. The above defined configuration has been used for each load balancing policy one by one and depending on that the result calculated for the metrics like response time, request processing time and cost in fulfilling the request has been shown. Parameters like average response time, data center service time and total cost of different data centers have taken for analysis.

Table 2: Comparison among load balancing policies (Average response time)

UBS	RR	TR	FCFS	ESCEL
UB1(1000)	50.64	50.65	50.63	50.64
UB2(2000)	51.16	51.15	51.15	51.18
UB3(3000)	51.85	51.82	51.85	51.83
UB4(4000)	52.48	52.49	52.48	52.47
UB5(5000)	301.49	301.56	301.54	301.57
UB6(6000)	200.91	200.90	200.89	200.90

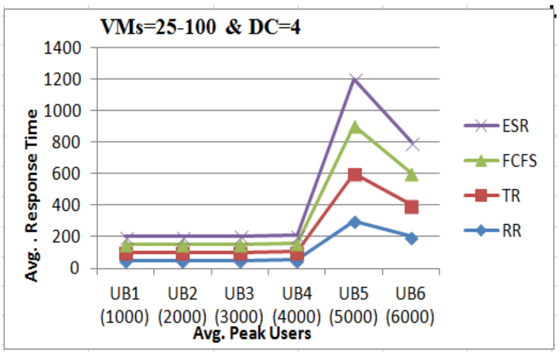

Fig 6: Result showing the average peak users vs. average response time

Table 3: Average data center request servicing time among LB policies

DCs	RR	RR	FCFS	ESCEL
DC1	0.786	0.785	0.783	0.784
DC2	1.567	1.574	1.578	1.577
DC3	2.1	2.1	2.093	2.097
DC4	2.773	2.769	2.775	2.77

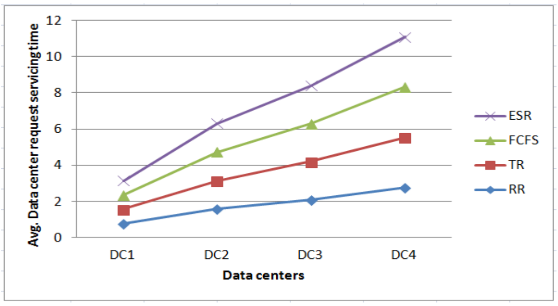

Fig 7: Analysis by taking average data center serving time and data centers

Table 4: Analysis of total cost

DCS	RR	TR	ESCEL	FCFS
DC1	2.64	2.64	2.64	2.64
DC2	5.08	5.08	5.08	5.08
DC3	7.65	7.65	7.65	7.65
DC4	10.10	10.10	10.10	10.10

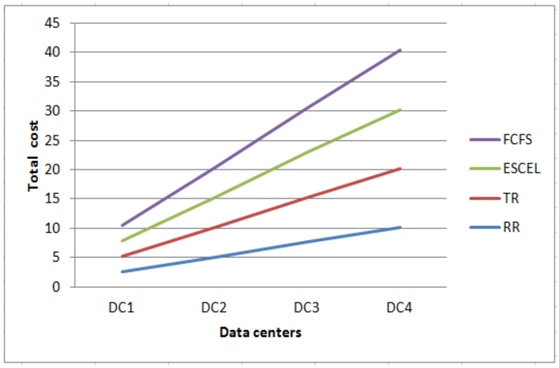

Fig 8: Total cost of different data centers

6. CONCLUSION

We have simulated four different scheduling algorithms have for executing the user request in cloud environment. Each algorithm is observed and their scheduling criteria like average response time, data center service time and total cost of different data centers are found. According to the experiment and analysis round robin algorithm has the best integrate performance. Future work can be based on this algorithm modified and implemented for real time system. Better response time can be expected if we apply some evolutionary algorithms such as PSO, ACO, and ABC instead of classical algorithms.

7. REFERENCES

[1] G. Pallis, “Cloud Computing: The New Frontier of Internet Computing”, IEEE Journal of Internet Computing, Vol. 14, No. 5, September/October 2010, pages 70-73.

[2] Qi Zhang, Lu Cheng, Raouf Boutaba, “cloud computing: state of-the-art and research challenges”, 20th April 2010, Spinger, pp. 7-18.

[3] M. D. Dikaiakos, G. Pallis, D. Katsa, P. Mehra, and A. Vakali, “Cloud Computing: Distributed Internet Computing for IT and Scientific Research”, IEEE Journal of Internet Computing, Vol. 13, No. 5, September/October 2009, pages 10-13.

[4] A. Khiyaita, H. EI Bakkli, M. Zbakh ,Dafir EI Kettani,” Load Balancing Cloud Computing: State Of Art”, 2010,IEEE.

[5] Ram Prassd Pandhy (107CS046), P Goutam Prasad rao (107CS039). “Load balancing in cloud computing system” Department of computer science and engineering National Institute of Technology Rourkela, Rourkela-769008, Orissa, India May-2011.

[6] J. Sahoo, S. Mohapatra and R. lath “Virtualization: A survey on concepts, taxonomy and associated security issues” computer and network technology (ICCNT), IEEE, pp. 222-226. April 2010.

[7] Bhaskar. R, Deepu.S. R and Dr.B. S. Shylaja “Dynamic Allocation Method For Efficient Load Balancing In Virtual Machines For Cloud Computing Environment” September 2012.

[8] R.Shimonski. Windows 2000 & Windows server 2003 clustering and load balancing. Emeryville. McGraw-Hill Professional publishing,CA,USA(2003), p 2,2003.

[9] R.X.T. and X. F.Z..A load balancing strategy based on the combination of static and dynamic, in database technology and applications (DBTA), 2010 2nd international workshops, (2010), pp. 1-4.

[10] Wenzheng Li, Hongyan Shi “Dynamic Load Balancing Algorithm Based on FCFS” IEEE, 2009. pp.1528-1531.

[11] Jiyni Li, Meikang Qui, Jain-Wei Niu, Yuchen, Zhong Ming “Adaptive resource allocation for preemptable jobs in cloud system”. IEEEInternational Conference on intelligent system design and applications, pp. 31-36, 2010.

[12] M Randles, D. Lamb, and A. Taleb-Bendiab, “A comparative study into distributed load balancing algorithms for cloud computing,” 2010 IEEE 24th international conference on advanced information networking and application workshops,2010, pp. 551-556.

[13] Jaspreet Kaur “Comparision Load Balancing Algorithms In A Cloud” International Journal Of Engineering Research And Applications. pp. 1169-1173, 2012.

[14] Bhathiya, Wickremasinghe.”Cloud Analyst: A Cloud Sim-based Visual Modeller for Analysing Cloud Computing Environments and Applications”, 2010, IEEE.

[15] Dr. Hemant, S. Mahalle, Prof. Parag R. Kaveri and Dr. Vinay chavan, “Load Balancing On Cloud Data Centers”. International Journal of Advanced Research in Computer Science and Software Engineering, Jan-2013.

[16] A. Singh, P. Goyal, S. Batra : Anoptimized round robin scheduling algorithm for CPU scheduling, International journal of computer and electrical engineerig (IJCEE), vol. 2, No. 7, pp 2383- 2385, December, 2010.

[17] Tanvee Ahmed, Yogendra Singh “Analytic Study Of Load Balncing Techniques Using Tool Cloud Analyst” . International Journal Of Engineering Research And Applications. pp. 1027-1030, 2012.

[18] A. Khiyati, M. Zbakh, H. El Bakkali, D. El Kettani “Load Balancing Cloud Computing: State Of Art”IEEE, 2012.

[19] R. Buyya, R. Ranjan, and R. N. Calheiros, “Modeling and Simulation of Scalable Cloud Computing Environments and the CloudSim Toolkit: Challenges and Opportunities,” Proc. of the 7th High Performance Computing and Simulation Conference (HPCS 09), IEEE Computer Society, June 2009.

[20] http://www.cloudbus.org/cloudsim