International Journal of Engineering & Technology, 7 (4) (2018) 5777-5782

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET
doi: 10.14419/ijet.v7i4.21491
Research paper

Enhanced load aware weighted round robin algorithm in cloud

ThulasiRaman [1] *, Arokia Paul Rajan [1]

[1] Department of Computer Science, CHRIST (Deemed to be University), Bengaluru, India
**Corresponding author E-mail: thulasiraman@cs.christuniversity.in*

Abstract

In a large-scale distributed cloud network there are lots of factors which affect the performance of the distributed systems, among this load balancing scheduling has a huge impact. It is recommendable to have a load balancer which equally splits the workload among all the available servers, according to the required parameters. The parameters of each server or a request can be termed as heavy loads or light loads relative to one another. Therefore, in the cloud environment, we need to assess the server capacity to overcome the high traffic and balance the loads properly. Subsequently, there is a need of a dynamic load balancing algorithm which splits and distribute all the loads equally on the basis of different parameters of the servers. The main aim of this research work is to address these requirements by devising a dynamic load aware load balancer for a heterogenous cloud environment. This is used to determine the weights of each queue dynamically based on the current traffic characteristics and static weights assigned to each server. The aim is to improve the average throughput and also to reduce the packet loss in the cloud networks. The experimentation of the proposed algorithm is performed using a simulator and the simulation results prove that there is a better improvement in the performance of the load balancer and also improves the average throughput compared with the existing WRR.

Keywords: *Enhanced Load-Aware WRR; Dynamic Weighting; Virtual Machine; Cloud Analyst.*

1. Introduction

In the recent years, Cloud Computing has become more apparent and secured technologies in the world. Cloud Computing which is also known as soft computing is one of the most remarkable technology due to the various services offered by it. Cloud computing systems are more depended on the term "virtualization" which helps to deploy and test all the services easily on a single physical server [1].

The Cloud systems offer many services like platform, software and infrastructure which are available to the users. All the users can use these services by paying the necessary subscription. These services are known as Platform as a Service (Paas), Software as a Service (SaaS) and Infrastructure as a Service (IaaS) in industries. Cloud Computing aims to strengthen the next-generation data centers by providing maximum resources available at any point of time over the internet [2]. There are various issues in the cloud computing services, one of the major problems is Load Balancing. It helps in preventing the deadlocks and ensures all the servers are running properly in heavy traffic among the network [3].

In this generation people depend on cloud for everything. All their shopping and financial needs are fulfilled by the cloud services. Cloud computing is one of the highly secured technology and so people store all their information's in the cloud storage. After storing these details every user gets a unique login credentials through which they can login and use it every time when they wish to login. This cloud computing is broadly classified into three main types as public, private and hybrid. Based on the type each and every cloud system has some specific storage capacity. Later based on the usage, this storage facility can be extended [4].

1.1. Load balancing

In cloud environment, the load balancing plays the major role by distributing all the available workloads equally to the servers. The load balancing technologies are offered by many cloud providers. But mostly Amazon Web Services (AWS), Google, Microsoft Azure and Rackspace play a major part. All the Cloud Service Providers (CSP) use this load balancer in their own cloud environment to ensure that the service is traffic free. Therefore, in every cloud environment, the load balancers ensure that all the work loads are distributed equally across the available servers, thereby reducing the network traffic and increasing the response time for the users [5]. Load balancing is used to improve the performance of the servers by minimizing the overall response time and dividing the loads equally to all the available servers. The main goals of load balancing are [6]:

- Maximize the throughput
- Avoid crashing
- Minimize the response time
- Maximize the performance
- Job prioritization

Figure 1 represents the process of load balancing in cloud computing.

Copyright © 2018 ThulasiRaman, Arokia Paul Rajan. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

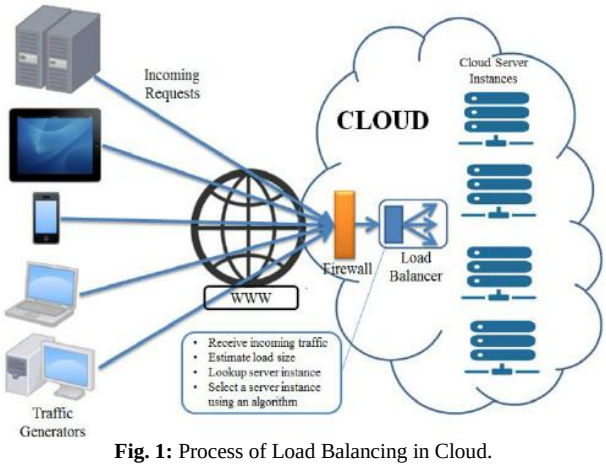


Fig. 1: Process of Load Balancing in Cloud.

Based on the uses, the load balancer can be classified as into two types as follows:

1.1.1. Static load balancer

In Static load balancing algorithm, the number of jobs is categorized and the behaviour of the servers are known. The decisions are made at compile time only and also the transfer decisions between the servers are independent of the actual machine state. In this technique, the functionality of the machines is determined at the beginning of the process and the works are assigned to the specific processors by the load balancers and are always processed by the same machine to which it is assigned. The main disadvantage of this static load balancer is that the host server is selected only after the requests are created and the fixed server cannot be changed until the request gets cleared [7].

1.1.2. Dynamic load balancer

In dynamic load balancing algorithm, the system makes use of the machine state information and allocates the jobs to the processor during the run time and this can also be changed according to the work load. Here the system monitors the processors all the time and whenever there is a load imbalance in the network, redistribution of the loads is done. In dynamic state, load balancer searches all servers in whole network and prefers only the server which has little load to balance the network traffic. Thus, a real communication process within the network should be made to choose the best servers and sometimes this may also result in the increase of traffic within the system [8].
The rest of the paper is organized as follows: Section II contains the related works, Section III contains the research problem statement, Section IV contains about the proposed model and the designed algorithm, Section V contains the experimental results and the performance evaluation comparing with the few existing algorithms, Section VI contains about the conclusion and the future enhancement works of this research paper.

2. Related work

The following survey represents the contributions that are influential in this research work:

2.1. Round robin (RR)

RR algorithm is one of the simplest algorithms which function on the principle of time quantum. The process of RR algorithm is done by randomly selecting the virtual machines. The request first hits the data center controller and then the datacentre controller assigns every request to the list of available VMs on a circular pattern. At first, a VM is selected randomly from the list and then the first request is allocated to it. Once the VM is allocated with a request, that VM is pushed at the last of the list. The main complication with this algorithm is that the advanced load balancing parameters like process times are not considered and also if the next VM is busy to response for the incoming job, then it is put in the wait queue [9].

2.2. Weighted round robin (WRR)

WRR algorithm was introduced to overcome the problems found in RR Scheduling such as priority scheduling and some of the problems mentioned above. The main principle of this algorithm is that every VM is fixed with some specific weight depending on which the requests are gathered, and the requests received are allotted to that particular weight. The WRR scheduling came into existence to better handle VMs with different parameters as per the capacity of individual VM. Generally, VMs with heavy weights receive the new requests first until the fixed integer value is attained and then the requests are moved to the other VM. Also, the VMs with equal weight, get equal requests in the load environment. The WRR scheduling is better than the RR scheduling when the principles and the functionalities of the VMs are different. But anyway, this also may lead to the load imbalance among the real servers in case of heavy weights, i.e. due to a static value to each server, first all the requests requiring immediate responses are directed to the same server at that time. The load here is not distributed evenly, which would lead to high efficiency and traffic between the VMs [10].

2.3. Load aware weighted round robin

The load aware WRR is the enhancement of weighted round robin algorithm. In this the shortcomings of the WRR principle is analysed and a solution is derived to overcome those problems. In WRR principle each server is fixed with some static weights and based on that weight the loads are assigned to it. In load aware WRR the static weights of all the server are identified at first so that all the servers can be balanced with equal loads. The load balancer serves each queue until the weight counter reaches zero. Since the weights of all the servers are known, no server can be left idle. But the datacentre sometime fails to record the status of all the available servers [11].

2.4. Equally spread current execution (ECSE)

In ESCE algorithm, the load balancer splits the requests equally to all the available VMs equally. The status of the VMs and the total number of requests assigned currently to the particular VM are available in the data center. So, if any new request needs to be assigned to a particular VM, it first looks into the index table to identify the machine which is loaded with the least number of requests. If there are more than one free VMs available, then the data center selects only the first free machine, then the load balancer also returns that VM id to the index table. The data center thus updates the index table by increasing a number. Also, when the VM completes a task, it is also identified by the index table and thus now it decreases the allocation count. Therefore, there is an additional computation done in this algorithm by scanning the queue again and again [12].
Figure 2 represents the mechanism of the load balancer by invoking any one of the load balancing algorithms in a cloud computing environment.

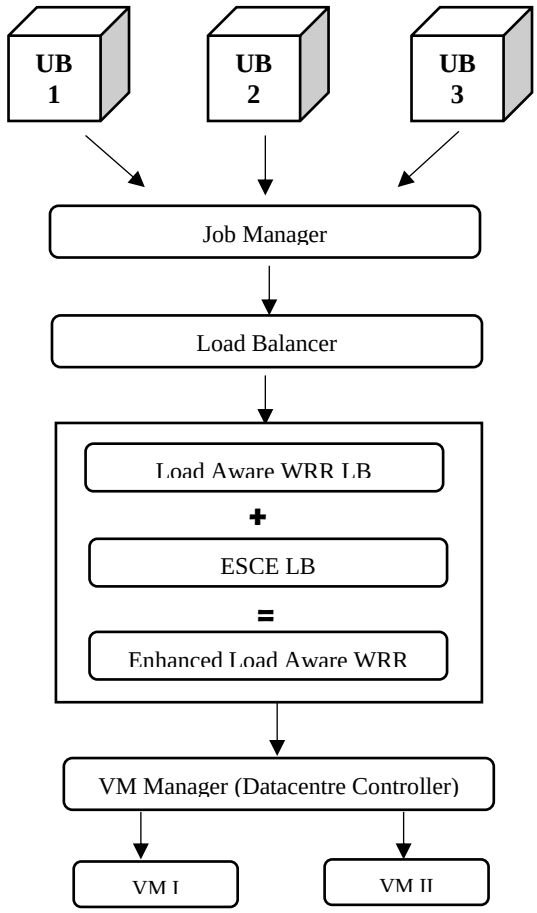


Fig. 2: Load Balancing Mechanism.

Thus, as shown in the above figure, a specific load balancing principle should be invoked for the process to split the loads equally to all the available servers.

3. Problem description

There is a set of services C (C_1, C_2, C_3…C_n). For each service C_n, a specific value V is assigned based on some criteria. These services are managed by highly configured servers S (S_1, S_2, S_3,….Sn) in the cloud environment. Each server Sn is characterized by its own capacity constraint. The state and functionality of each server is available in the Data center Controller (DC). At a time T, there comes a group of requests R (R_1, R_2, R_3….R_n) raised by the users, each request R_n is served by some specific service Sn. The load balancing principle is to split the loads equally across all the available servers corresponding to their fixed parameters and functioning according to the requests [13].

4. Proposed model

In this section, a new enhancement of the WRR scheduling algorithm, which we call Enhanced LAWRR, is described. First, we represent the shortcomings of the WRR scheduling principle. The WRR load balancer is the extension of the RR load balancer that serves the requests in relation to the fixed weights. According to the priority of the service classes the requests are identified. The weights of all the classes are determined on the basis of minimum required parameters. Each server is assigned to some fixed weights. The main role of WRR is to allocate the loads as per the fixed weights and then transfer the remaining loads to the other servers. In this case, only one server will be functioning at a particular time and all the loads will be assigned to that server until the fixed weight is reached. So, this leads to heavy traffic on the first server while the other server remains idle. Hence this also increases the delay in response as well as decreases the throughput. To overcome the shortcomings of WRR principle described above, an algorithm has been proposed by us which modifies the static weight into the dynamic weight. Therefore, the weights have been calculated dynamically according to the number of loads present in each queue and at the same the fixed static weights of each classes are also identified. These parameters have a control over the network traffics of each queue which will delay the entire process and packet loss in the network. On the other hand, this methodology also helps to improve the throughput and reduces the packet loss.

The solution is designed based on the following assumptions: Some of the servers are assigned with some specific weight, the server which is assigned with heavy weights are assigned first and then it moves to the next server. Till that time the remaining server remains idle and all the works are performed by the same server which maximize the process time. Each request in the queue has its own memory needs and service time. So here we are combining the ECSE load balancer with WRR load balancer to make the process as dynamic. So, the state of the server is monitored continuously and the requests are split equally across all the servers in cloud environment.

The dynamic method of calculating the weight of the VMs is a two-step process. At First the availability of all the servers is checked in the DCs and then the fixed weight of each server is determined. Then on combining ECSE load balancer with the existing WRR load balancer we can split the loads equally to available servers. Thus, this ensures that no server remains idle and all the requests are served immediately.

4.1. Proposed algorithm

Algorithm 1: Enhanced LAWRR scheduling

```
N ← Total number of requests in the queue
q_{j,m} ← current request of queue j at round m
w_{j,m} ← dynamic weight capacity of queue j at round m
WE_{j,m} ← weight counter of queue j round m
m ← current RR round
WE_{j,m} ← 0 (j = 0, 1, 2, ... , N − 1)
m ← 0
for j ← 0 to N − 1 do
 if q_{j,m} ≠ NULL then
 compute w_{j,m} using Equation 6
 WE_{j,m} ← W_{j,m}
 if q_{j,m} ≠ NULL and WE_{j,m} ≠ NULL then
 transmit packets from q_{j,m} using EWRR
 else
 m ← m + 1
 End if
end
```

Therefore, the static weights of the server are changed into dynamic. So now if a queue is picked for a service first it checks for the data center controller to check the status of all the available servers and then the fixed weights of the server are also identified. Based on the total number of requests, it is splitted equally to all the available servers using the Equation 1 mentioned below:

$$W_i = \frac{MRTRi}{\sum_{i=1}^{N} MRTRi} \quad (1)$$

Here W_i represents the weight of the queue i, MRTRi represents the rate of traffic in the queue, N represents the total number of the requests in the queue.

On computing the loads based on the above equation the LAWRR scheduler has modified the static weights of the queue, say q_1 and q_2 dynamically according to the traffic present in the queue. After fixing the dynamic weights to the weight counter, the proposed dynamic load balancer starts providing response to all the available loads in the queue starting from the first to the end of the queue. Likewise, it servers to provide service to all the available loads in the queue in a circular pattern until the queue becomes empty or either the weight count in the counter becomes zero. Then at last

the counter is reset or the load aware scheduling principle process is terminated.

5. Experiment results and discussion

The simulation process for the proposed scheduling principle is done using cloud analyst tool and the results have compared to prove the effectiveness of the proposed methodology.

5.1. Cloud analyst

There are many simulation tools that can be used to experiment a model of simulated environment. But it is necessary that having a tool in which the results can be visualized easily and will be helpful to compare the results later. Such a tool separates the simulation environment and enables the modeler to focus on the various functions of the simulator rather than the technical functionalities of program. It also helps the application to implement the simulation process continuously with different modifications to the required parameters easily. A graphical representation of the simulation results will be useful in analysing and comparing the performance of the simulation with the existing ones. The Figure 3 represents the architecture of the cloud analyst.

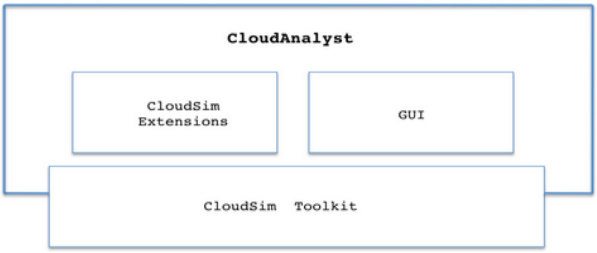


Fig. 3: Cloud Analyst Architecture

It is clearly seen that cloud analyst is built on the top on the cloudsim tool kit, by extending cloudsim functionality with the introduction of GUI parts to view and compare the results for a better clarification. Another one main purpose to invent this cloud simulator is that all the results can be saved in the pdf format and it can also be visualized in the form of bar diagrams and charts which may help the users to analyse and compare it for further process [14].
The main components of cloud analyst are as follows:
User Base: It generates the traffic and is used to assign various group of users that can be treated as a single user base in the simulation environment.
DataCeter Controller: It contains all the information and status of all the available servers
GUI packages: This is mainly useful to represent the application with various graphical formats.
Internet: The internet traffic routing is created by this function by integrating the transmissions and data transfer process
VMLoadBalancer: This is used by the data center controller to identify which VM should be allocated to the next requests in the queue
Simulation: Simulation is the main concept of the process which accepts the requests from the user base and then executes those requests
CloudAppServiceBroker: The traffic routing between the data center controller and the user base are handled by this component [15].
The Figure 4 represents the dashboard of the cloud analyst simulator.

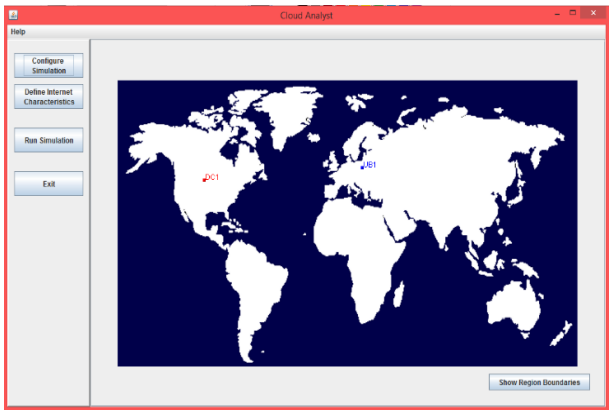


Fig. 4: Cloud Analyst Dashboard.

The simulator for the load balancing policies are configured to analyse the parameters in the cloud analyst. We have configured the parameters for the application deployment, such as user base and data center configuration which is shown in Figure 5.

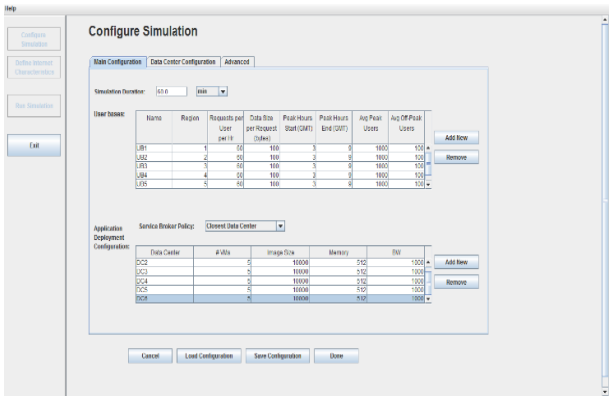


Fig. 5: User Base and Data Center Configurations.

After configuring the user bases and the data centers for all the particular region, the principle how the load balancer should function needs to be selected. Every load balancer has a unique load balancing policy based on which it performs the scheduled operation. This simulator can have maximum number of load balancing algorithms invoked in it. But the only drawback with this simulator is that every time when a user selects the algorithm, the application should be compiled and started from the beginning. The selection of load balancing policy, which is the main part of the simulation is shown in the Figure 6. Based on this policy only the load balancer splits the load and maintains the traffic [16].

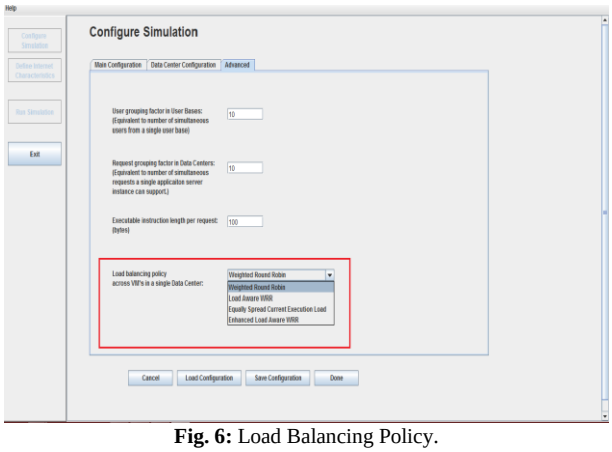


Fig. 6: Load Balancing Policy.

Now the various user bases in different regions have been defined and configured. we consider five data centers to serve the users requests. First DC is assigned in the region 3, second in the region 1, third in the region 4, and fourth DC is located in region 2. There are

totally 30 VMs in each of DC1 to DC5. Then after configuring these parameters, the load balancing policy should be selected. The duration of the simulation also can be defined. After this, you can run the simulation from the dashboard. Now the cloud analyst runs the simulation process for different parameters repeatedly [17]. Experiments were conducted for different configurations and the response time according to the different user bases are tabulated in the Table 1.

Table 1: Response Time of different User Bases

User Base	WRR- Avg Time (ms)	ECSE- Avg Time (ms)	Enhanced WRR- Avg Time (ms)
UB1	50.25	50.10	49.83
UB2	49.94	50.12	48.80
UB3	50.01	51.15	49.12
UB4	50.26	50.25	49.30
UB5	51.12	51.11	50.25
UB6	51.60	50.7	50.10

The simulation results are obtained and the comparison chart for all the three load balancing principles are plotted for the first three user bases as shown in the Figure 7.

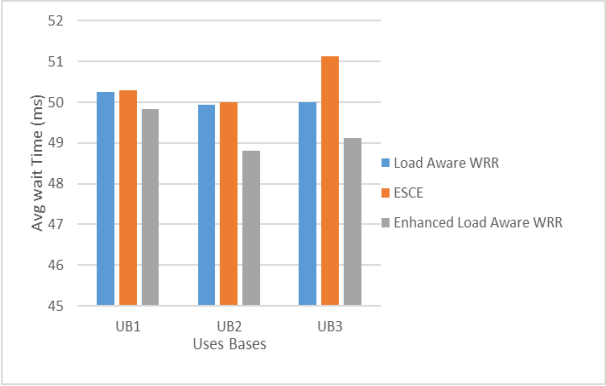


Fig. 7: Average Response Time for the Three Principles.

Next the response time between the datacentre and the VMs are also obtained separately and the processing time values are tabulated as shown in the Table 2.

Table 2: Datacenter Processing Time

Data Center	WRR-Avg Time (ms)	ECSE- Avg Time(ms)	Enhanced WRR-Avg Time(ms)
DC1	0.49	0.50	0.48
DC2	0.47	0.48	0.47
DC3	0.48	0.47	0.47
DC4	0.45	0.45	0.44
DC5	0.50	0.51	0.49
DC6	0.51	0.49	0.50

The simulation results are obtained and the comparison chart for all the three load balancing principles are plotted for the first three user bases as shown in the Figure 8.

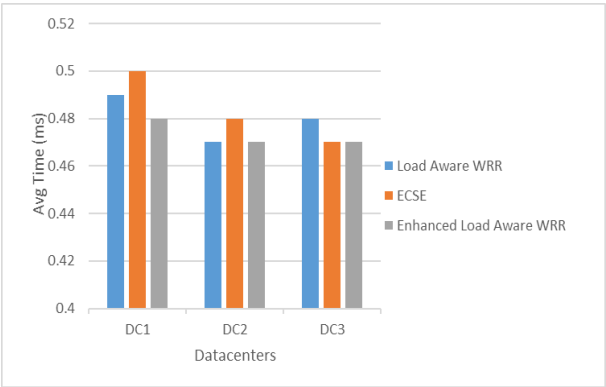


Fig. 8: Average Response Time for the Three Principles.

Thus, when comparing the response time with all the three existing load balancing principles, the proposed model shows a efficient change in the response time and also there is a change in the processing time between the data centers.

6. Conclusion

The overall objective of this research is to design a dynamic load balancing principle that allocates the requests dynamically depending on the different attributes of the servers and the fixed weights. The first objective of this research was achieved by introducing a dynamic method to analyse and find the most preferred attributes of the servers. Here a new enhanced approach had been presented to resolve the burstiness of the traffic congestions among the cloud network. The second objective of this research is achieved by identifying the fixed static weights and distributing the loads equally to all the available servers. Experiments were done using a cloud simulator and the results prove that the proposed algorithm exceeds the existing ones in terms of packet loss and at the same time it increases the average throughput. The designed dynamic principle is found to be suitable for managing loads in a large-scale cloud network. Extending this dynamic solution for all the cloud models is a desirable future enhancement of this research.

References

[1] Reena Panwar and Bhawna Mallick , "Load Balancing in Cloud Computing using Dynamic Load Management Algorithm", 2015 International Conference on Green Computing and Internet of Things (ICGCIoT), https://doi.org/10.1109/ICGCIoT.2015.7380567.

[2] Rodrigo N. Calheiros, Rajiv Ranjan, Anton Beloglazov, Cesar A. F. De Rose and Rajkumar Buyya , "CloudSim: a toolkit for modelling and simulation of cloud computing environments and evaluation of resource provisioning algorithms", Wiley Online Library in 2010.

[3] Muskan Garg and Rajnesh Narula, "Optimized Load Balancing in Cloud Computing using Hybrid Approach of Round Robin and Max-Min Scheduling", International Journal of Science and Research, Volume 6 Issue 1, January 2017, ISSN (Online): 2319-7064.

[4] Brototi Mondal, Kousik Dasgupta and Paramartha Dutta, "Load Balancing In Cloud Computing using Stochastic Hill Climbig-A Soft Computing Approach", Procedia Techology-sciverse ScienceDirect, https://doi.org/10.1016/j.protcy.2012.05.128.

[5] Suman Rani, Vinod Saroha and Sanjeev Rana, "A Hybrid approach of Round Robin, Throttle & Equally Spaced Technique for Load Balancing in cloud environment", International Journal of Innovations and Advancement in computer science, Volume 6, Issue 8, August 2017, ISSN 2347 – 8616.

[6] Ramesh Prajapati, Dushyantsinh and Samrat Khanna, "Comparison of static and dynamic load balancing in Grid Computing", International Journal for Technological Research in Engineering, Volume 2, Issue 7, March-2015, ISSN (Online): 2347 – 4718.

[7] Zenon Chaczko, Venkatesh Mahadevan, Shahrzad Aslanzadeh, and Christopher Mcdermid, "Availability and Load Balancing in Cloud Computing", International Conference on computer and software modelling, IPCSIT vol. 14-2011.

[8] Ahmed Alsheikhy, Reda Ammar and Raafat Elfouly, "An Improved Dynamic Round Robin Scheduling Algorithm based on a variant quantum time", International Journal of computer science, Engineering and Information Technology, Vol 5, No.1 February 2016.

[9] Abdulaziz Alnowiser, Eman Aldhahri and Abdulrahman Alahmadi, "Enhanced Weighted Round Robin (EWRR) Scheduling with DVFS Technology in Cloud", in Intenational Conference on Computational Sciene and Computational Intelligence, https://doi.org/10.1109/CSCI.2014.62.

[10] Vishwas Bagwaiys and Sandeep k. Raghuwanshi, "Hybrid Approach using Throttled and ESCE load balancing algorithms in cloud computing", International journal of web and semantic technology, April 2016, vol 3. Issue 2, P33, https://doi.org/10.1109/ICGCCEE.2014.6921418.

[11] R.Arokia Paul Rajan, "Service Request Scheduling based on Quantification Principle using Conjoint Analysis and Z-score in Cloud", International Journal of Electrical and Computer Engineering (IJECE), Vol. 8, No. 2, April 2018, pp . 1238~1246.

[12] Kalpana Ettikyala and Y. Rama Devi, "A Study on Cloud Simulation Tools", International journal of computer applications, vol 115-No. 14, April 2015.
[13] Harlen Kaur and Er. Vinay Gautam, "A survey of various cloud simulators", International journal of computer sciences and engineering, vol-2, Issue-9, 2016.
[14] Jayaprakash Maltare and Balwant Prajapat, "Dynamic Load Balancing in cloud computing using cloud sim", International Journal of Computer applications, Vol 148-No.5, August 2016.
[15] Abhay Kumar Agarwal and Atul Raj, "A New Static Load Balancing Algorithm in Cloud Computing", International Journal of Computer Applications, Volume 132-No.2, December 2015.
[16] Seema Nagar and Ajeet KR Bhartee, "A comparative study on load balancing algorithms in cloud computing", International journal of computer science trends and technology (IJCST), vol-3, Issue-4, Aug 2016.
[17] Sunita Rani Jindal and Sahil Vashist, "A sophisticated study of round robin and equally spread current execution in cloud computing", International journal of advanced research in computer science and software engineering, vol-4, Issue-8, Aug 2014.