Cloud Publications
International Journal of Advanced Computer Science and Information Technology
2016, Volume 5, Issue 1, pp. 83-100, Article ID Tech-646
ISSN 2320 – 0235

Research Article **Open Access**

Analysis of Different Methods for Measuring the Performance of Database on Cloud Environment

Akrati Sharma and Sanjiv Sharma

Department of CSE/IT, MITS, Gwalior, M.P., India

Publication Date: 22 July 2016

Article Link: http://technical.cloud-journals.com/index.php/IJACSIT/article/view/Tech-646

Abstract Cloud computing become most adoptable technology over the past few years, it is widespread for both at organizational level or for the person who use the services that are offered in the cloud. There are several technocrats currently researched on cloud related issues and soon cloud represents the modern computing. The cloud gave the prospect for the enterprise and allowing them to center on their work by providing hardware and software solution without developing them own. Now a day the database has also moved to cloud computing so we will look into the fine points of database as a service and it's servicing. While storing data on cloud there is a need to balance the load on datacenters because if user base always sends request on single data center then it becomes overloaded so load balancing techniques are applied to manage this problem. To manage geographic distribution in terms of computing servers and data workloads a tool termed as CloudAnalyst is used which is based on cloudsim technique. CloudAnalyst helps developers with the vision of distributing applications among cloud infrastructures. Currently tool having three algorithms for load balancing round robin, throttled load balancer, equally spread current execution and the proposed algorithm is weighted round robin which works better as comparison to round robin in various aspects.
Keywords *Data Center; User Base; CloudSim; CloudAnalyst; Load Balancing; Round Robin; Throttled Load Balancer; Equally Spread Current Execution; Weighted Round Robin*

1. Introduction

Cloud computing is a field which brings a boom in the industry of technology and academics. The dependency on cloud in both the areas increases day by day. The main reasons for this popularity are the unique features and services provided by the cloud. It offers distributed services, usage of virtualized resources, sustain full realization of computing as a utility in the future [1]. With the popularity of the cloud technology, several new potentials for internet based applications become cynosure. The application models can be classified on the basis of parties who use this technology. First one is the vendors or cloud service providers who provide large scale computing infrastructure at low budget. Second one is the large-scale software systems providers, who manage large scale applications such as social networking sites and e-commerce.

These cloud services minimize costs and improve service quality to end users. With the beginning of the cloud, deployment and hosting became easy and less expensive credit goes to the functionalities of cloud like pay per usage, flexible infrastructure services, on demand self-services and many more. As per in previous era the development of such applications gaining requirement of servers with a predetermined capacity and able to handle the expected load at peak demand, installation of software infrastructure and the platform supporting the same applications. All because of this the servers were underutilized and the reason is the peak traffic occurs occasionally or at specific short time periods.

When the above two ends are taken into account, several factors generates that affect the net assistance of cloud. These factors include distribution of user bases at different geographical areas, performance of the internet infrastructure in those geographic areas, working of user bases, and capabilities of cloud services and its behavior towards dynamic reconfiguration among others. To understand such vibrant and largely distributed environments in a controlled and reproducible manner there is a need of simulation.

In Database as a service [7] a database can be accessed by the clients via internet from the cloud. Database service provider deliver database to them when they need it or want to store their data over cloud. Cloud database is designed for creating the pool of virtualized resources so that user can select the required resource and used it. The cloud database is present on cloud so that it uses the various services of cloud and utilizing the software and hardware resources of the cloud computing service provider. The cloud database holds the data on heterogeneous data centers which is located at heterogeneous locations. This is the reason which makes cloud database structure different from the normal databases, due to this its structure become complex one. There are numerous nodes across the cloud databases for query services, so that data centers that are located in different geological areas can also be accessible. This linking of nodes on datacenters is compulsory for the access of the database over the cloud. There are several methods for accessing the database over the cloud; the user can access it via computer through the internet using 3G or 4G services.

2. CloudSim

CloudSim is a framework developed by the GRIDS [3] laboratory of University of Melbourne. A widespread and extensible simulation framework based on java allows flawless simulation, and perform experiment on emerging cloud computing infrastructures and application services. CloudSim [2] is just like a boon for researchers and industry based developers using this they can work on specific system design issues without worrying about the low level details of cloud based infrastructures and its services.

CloudSim consist the following features:

1) Allow modeling and simulation for large scale cloud computing infrastructure, having data centers on a single node.
2) Provide platform for modeling data centers, service brokers, scheduling, and allocations policies.
3) Presence of virtualization engine, which helps in creating and managing multiple, self-employed, and co-hosted virtualized services on a data center node.
4) Flexibility to switch between spaces and time-shared allocation of processing cores to virtualized services.

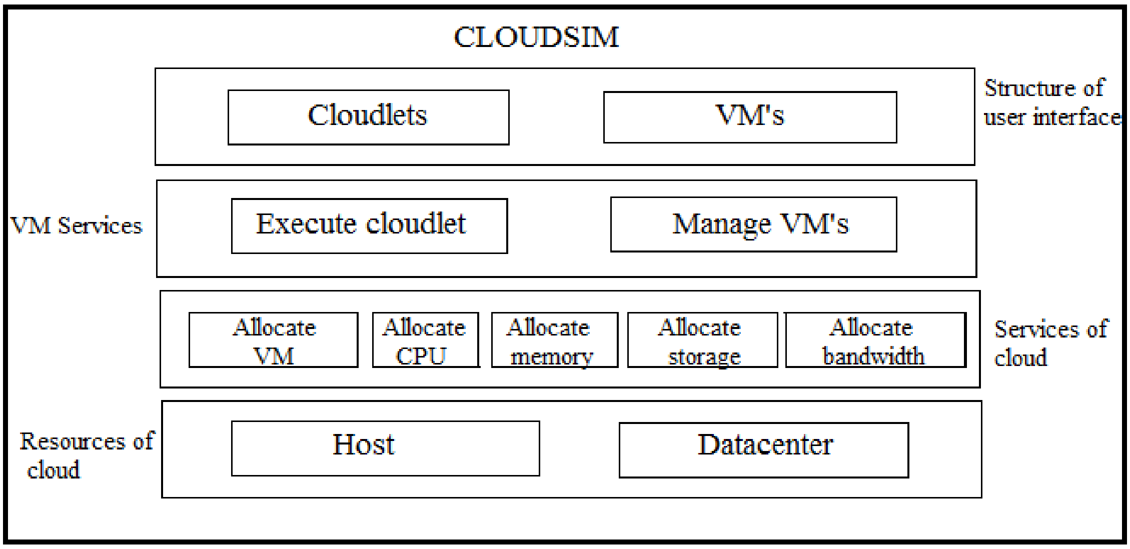

Figure 1: *Components of cloudsim*

These convincing features of Cloudsim would pace the formation of new resource allocation policies and scheduling algorithms for cloud computing. CloudSim framework developed on the top of GridSim framework which is also developed by the GRIDS laboratory.

2.1. GridSim

GridSim [4] toolkit was developed to rectify the problem of performance evaluation related to real large scaled distributed environments in a repeatable and restricted manner. The GridSim is a Java based simulation toolkit which supports modeling and simulation of diverse grid resources. It supports multiple application models and provides some rules for creating application tasks, mapping of tasks to resources.

2.2. SimJava

SimJava [5] is the simulation toolkit which is incorporated in both the simulation framework CloudSim and GridSim.

3. CloudAnalyst

It is an easy to use tool with visualization capability. CloudAnalyst [6] separates the simulation experiment and setting up a programming exercise and allow modeler to focus on the parameters of simulation more than the procedure of programming. It allows the repetition of simulations with changes according to the parameters. Output generates graphically which is helpful in terms of analyzing the results after the simulation process. If there is any problem with the parameters of simulation like performance evaluation and accuracy of result it also rectify the same. Figure 2 consist the architecture of the simulator. Some of the features of the tool describe it more deeply:

1) It is easy to set up and execute a simulation experiment in the environment. The simulator provides an easy to use GUI which is spontaneous & inclusive.
2) The tool is easy to configurable, especially in modeling something as complex as an internet application which rely on parameters and its values. Hence there is need of rapidly changes the values of parameters according to simulation and repeat them.

3) Specialty of the tool is the representation of result in the forms of graphs, tables and charts. It is highly advantageous to summarize large amount of statistics that is collected during the simulation. It is helpful in finding the unique pattern between the output parameters and comparing them.
4) Repeatability of experiments is the striking feature of the simulator. It is important that every time when the experiment is performed it must produce same result for same parameters and different results for different ones. Otherwise the simulation becomes messy and becomes just a random sequence of events rather than a controlled experiment. There is also an option of saving the old simulation and if required in future then loaded back from the directory.
5) It is realistic simulation framework incorporated with the set of input parameters can be achieved in a few attempts. There is ease of extensions in terms of parameters on the simulator, the parameters can easily be increased or decreased according to the requirement of the simulation. Hence the simulator architecture supports extensions with minimum efforts.

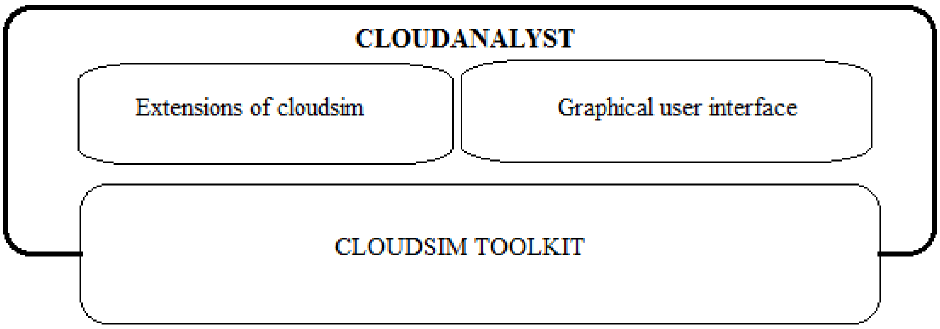

Figure 2: *CloudAnalyst architecture*

3.1. Simulation Output of the Simulator

When experiments are performed on the simulator the generated results can be calculated with following parameters:

1) Response time of the simulation, it refers to the amount of time server takes to return the results of a request to the user. It affects by the factors like network bandwidth, number of users, number and type of requests submitted, and calculation time. Average, maximum and minimum response time is calculated
2) Response time can be calculated on the basis of regions present across all over the world and the overall effect of that usage on the data centers hosting the application.
3) Data center servicing time is calculated, it is the taken by the server to fulfill the request of the user.
4) Data center processing time is calculated of the overall simulation.
5) Cost of the simulation is calculated.
6) Time to transfer data is also taken into account.

CloudAnalyst is purely java based tool and on Java platform it using Java SE 1.6. It is designed on JavaSwing the GUI of the simulator using Swing components.

3.2. Load Balancing

As described in the figure that there are several datacenters (DC) and user bases (UB) are located across the world UB's are requesting for storing their data on the DC's. Now the question is arises that on which data center the requirements of users are fulfilled. Here some conditions are occurred which is undesirable:

1) If user always sends data to a data center which is filled with request and still receiving much requests then the problem of overloading occurred due to which data centers crashed.
2) If requests always arrived on single data center then other data centers remains idle.

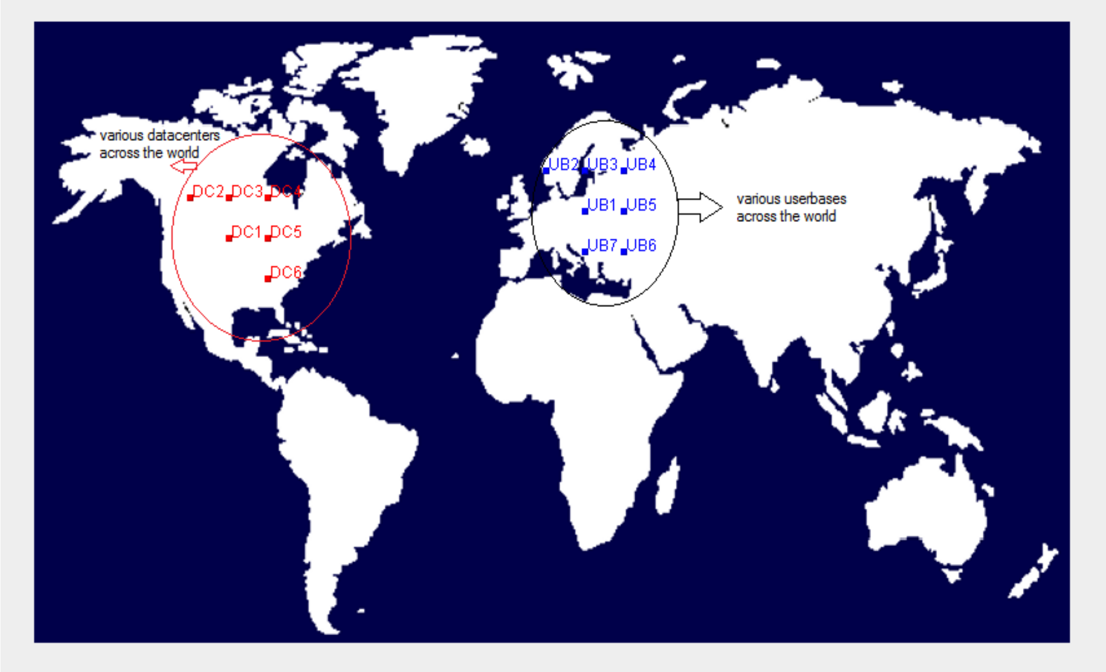

Figure 3: *World map having several data centers & user bases*

To solve such problems the concept of load balancing is used. Load Balancing is a method to distribute workload on the multiple data centers or a computer cluster through network links to achieve optimal resource utilization for increasing throughput and lower down the response time. Load Balancing is used for avoiding too much overload on the resources and dividing the traffic between different data centers. Data can be sent and received without maximum delay. Load Balancing is used for lower down the total waiting time of the resources. In cloud computing load balancing are uses for maintaining the load on virtual machine and cloud resources. To perform the task of load balancing on cloud data centers a simulation environment is required that is CloudAnalyst simulator. The tool is java based and based on cloudsim.

4. Backgrounds & Related Work

Sivashakthi, T., Dr. Prabakaran N., [8] in 2010 gives various storage techniques for storing data in cloud. Cloud storage is regarded as a system of distributed data centers that generally utilizes virtualization technology and supplies interface for data storage. The paper includes discussion on: Implicit storage security to data in online, Public auditing with complete data dynamic support, efficient third party auditing (TPA), Way of dynamically store data in cloud, Effective and secure storage protocol. These points are described in details.

Nusrat Pasha, Dr. Amit Agarwal, Dr. Ravi Rastogi [9] in 2014 explains the working of round robin algorithm. It is the scheduling technique that uses the concept of time slices. The time is divided into multiple time slices and each node is given a specific time slice which uses the principle of time scheduling. The resources provided to the requesting client on the basis of time slice by the of the service provider. Round robin algorithm also is incorporated in cloud databases for performing the task of load balancing on different data centers of cloud. The Round Robin algorithm does not save the state of previous allocation of a VM to a request from a given user base while the same state is saved in RR VM load balancer. The Round Robin VM Load balancer maintains two data structure. Hash Map- in which it stores the entry for the last VM allocated to a request from a given user base. VM State List- this stores the allocation status (i.e. busy available) of each VM.

Waleed Al. Shehri [10] in 2013 gives a brief about DBaaS means database-as-a-service. This is a storage technique for storing databases on cloud. A database can be accessed by the users through internet and they can use it according to their requirement. Cloud database is designed for virtualized computing environment. The cloud database is incorporated with the help of cloud computing that means using the software and hardware resources of the cloud service provider. The cloud database will become the mostly used technology for storing huge data worldwide. It is not like taking the relational database and deploys it over a cloud server. It means that adding of extra nodes according to user requirement, and maximizes database performance. It is required to distribute the data over different data centers. The database must be available 24x7 so that the user can get the data whenever he needs. The cloud database must be easily manageable and it should be less costly too. Cloud computing is helpful in recovering the information after a disaster like crashing of nodes in the database.

Gill Sukhjinder Singh, Thapar Vivek [11] in 2015 gives a brief about the load balancing and its need on cloud environment. Load Balancing is a procedure to distribute or share the load equally among all the nodes (in this case we considered it as datacenters in place of nodes) of the network. It helps the congested and under loaded nodes like if any node is having more loads (congested node) than the threshold value, then its load is transferred to the node with less loads. Thus load balancing is the best choice for performing this operation because it is a major challenge for cloud computing. Several time it may inevitable and not affordable, if any of the servers become idle in a datacenter while others are overloaded or congested in response to user demands. It means jobs require the proper assignment of servers in a datacenter and results is the higher maintenance cost of idle servers. Therefore appropriate load balancing techniques need to be applied for extra cost effective in the cloud environment.

Bhathiya Wickremasinghe [12] in 2009 presented a tool termed as CloudAnalyst used for analyzing cloud environment and it is completely based on the cloudsim technology. CloudAnalyst is a trouble-free tool with a level of visualization capability is even better than just a toolkit because it visualizes the whole world map and shows the geographic locations of data centers and user bases over there. Such a tool separates the simulation experiment set for programming exercise and enables a modeler to give attention on the simulation parameters rather than the technicalities of programming. It allows performing the experimentation process repeatedly with some small modifications to the parameters rapidly and easily. The result generated in graphical form so that result analysis becomes easier and more efficient and it enlightened the problems with the performance and accuracy of the simulation logic.

5. Existing Algorithms Work on Simulator

There are some algorithms which work on CloudAnalyst tool to perform load balancing. Figure 4 shows the options of such algorithms in the tool and its description are as follows:-

1) Round robin policy

It is one of the simplest scheduling techniques [13] that utilize the principle of time slices. Here the time is allocated in the form of slots and hand over it to the each node; it is the concept of time scheduling. Each node is given a time quantum and its operation, the resources or virtual machines are provided to the requesting client on the basis of time slice by the cloud service providers.

2) Throttled load balancer policy

This algorithm assures that pre- defined numbers of cloudlets are allocated to a single virtual machine at a time instant. If the number of users requests are more than that of the present number of available virtual machines at data centre. Than incoming request are arranged in the queue basis until the next virtual machines becomes available.

3) Equally spread current execution load policy

The jobs are submitted by the clients to the computing system. As the submitted jobs arrive to the cloud they are queued in the stack. The cloud manager estimates the job size and checks for the availability of the virtual machine and also the capacity of the virtual machine. Once the job size and the available resource (virtual machine) size match, the job scheduler immediately allocates the identified resource to the job in queue. Unlike the round robin scheduling algorithm, there is no overhead of fixing the time slots to schedule the jobs in a periodic way. The impact of the ESCE algorithm is that there is an improvement in response time and the processing time. The jobs are equally spread, the complete computing system is load balanced and no virtual machines are underutilized. Due to this advantage, there is reduce in the virtual machine cost and the data transfer cost.

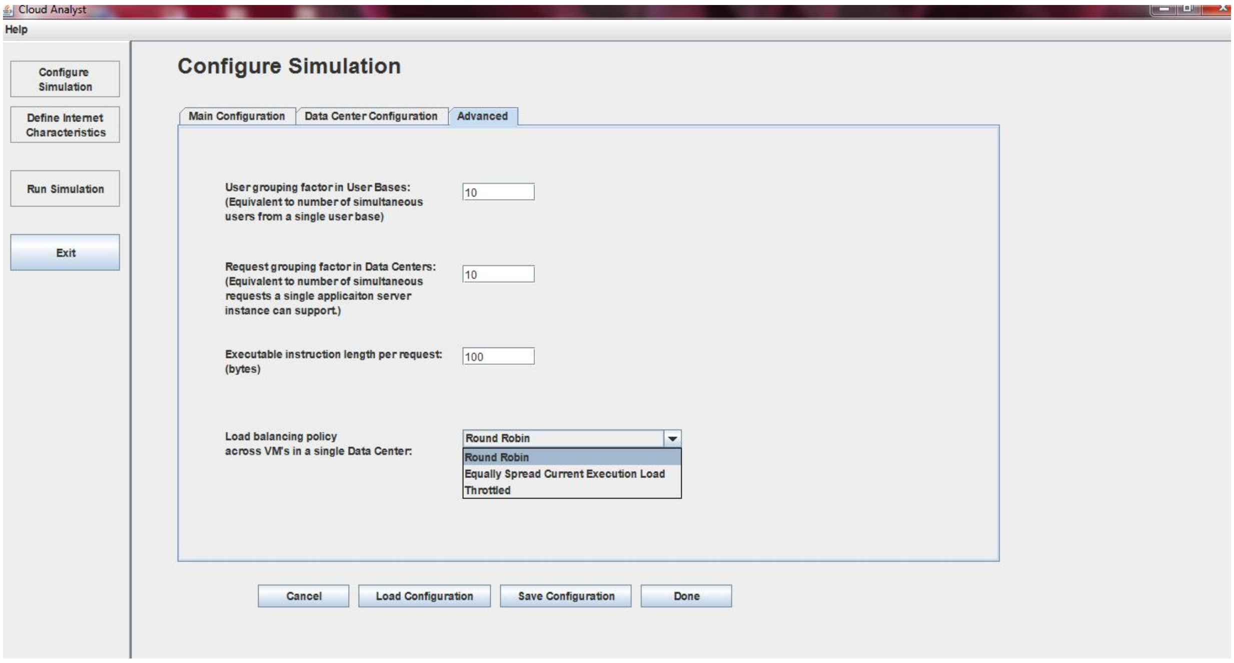

Figure 4: *CloudAnalyst screen shows existing load balancing algorithm*

5.1. Proposed Algorithm for Performing Load Balancing

For performing load balancing besides these algorithms one more algorithm is used which is the enhanced version of round robin algorithm that is the weighted round robin algorithm. Weighted round robin perform better load balancing as compare to round robin policy. Figure 5 shows the incorporated option of weighted round robin algorithm in the tool. In round robin policy the virtual machines (VM) are allocated to the data centers for particular time slice. If there are total 5 data centers then 2ns (approx.) time slice is allocated to every data center. It means that every data center uses VM for 2ns; it doesn't matters that whether any data center is needy for VM. This problem is solved in weighted round robin algorithm, in weighted round robin algorithm a weight is assigned to every data center; more weight shows more requirement of VM. The data center having more weight the VM first allocate to that data center.

In terms of load sharing we can say that suppose there are 5 data centers and user want to store its data on data center then using weighted round robin policy a weight is assigned to each data center. Weight is decided on the basis of request arrives on the data center, like if a data center DC1 filled with the request and it does not have enough capacity to handle more request then a weight assigned to DC1 is less and another data center DC2 having enough space to handle request than its weight is more as compare to DC1 hence there are more chances that next request arrives on DC2 due to its weight. It decreases the chances of overloading and crashing of data centers

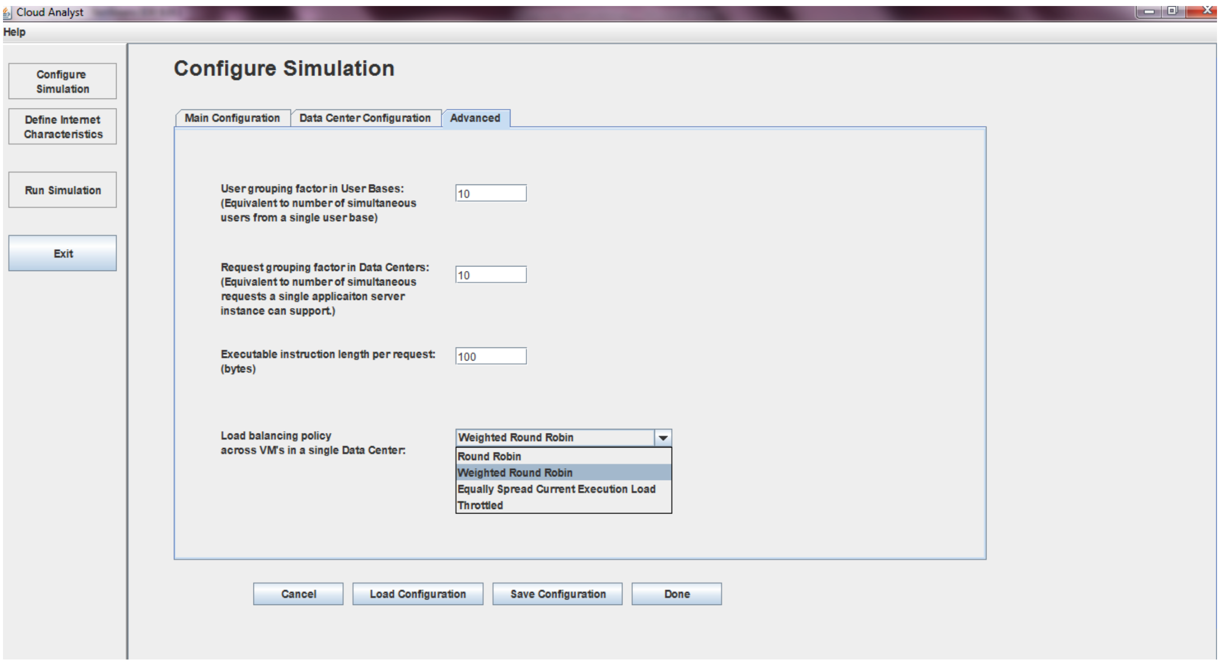

Figure 5: *CloudAnalyst screen shows proposed load balancing algorithm*

6. Result Analysis

There are several measures on the basis of which it is proved that the outcome generated through weighted round robin policy is better than that of round robin policy. Load balancing task can be performed in a better way via using weighted round robin algorithm.

6.1. Response Time

Response time refers to the amount of time server takes to return the results of a request to the user. It affects by the factors like network bandwidth, number of users, number and type of requests submitted, and calculation time.

$$\text{Tresponse} = ((n/r - T_{calculation}) / 1000) \text{ ms} \ldots\ldots\ldots (1)$$

n: number of concurrent users
r: number requests/sec the server receives
$T_{calculation}$: calculation time
With the response time various service broker policies are also taken into account:

The traffic routing between user bases and data centers is controlled by a service broker that decides which data center should examine the requests from each user base. The CloudAnalyst uses three types of service brokers which help in implementing the different routing policies.

6.2. Closest Data Center Policy

In this case one has to search quickest path to the data center from a user base according to network latency. The service broker will have to search the data center for the user which is at the least distance from particular user base and route user traffic to the closest data center according to transmission latency. Figure 6 shows the calculation of response time using closest data center policy in round robin algorithm whereas Figure 9 shows the same with weighted round robin policy which generates better results.

6.3. Optimized Response Time Policy

In this policy the service broker dynamically monitors the performance of all data centers and manages the traffic to the data center. It estimates to give the best response time to the user at the time when user quires. Figure 7 shows the calculation of response time using optimal response time policy in round robin algorithm whereas Figure 10 shows the same with weighted round robin policy which generates better results.

6.4. Reconfiguring Dynamically with Load Policy

This is an extended version of proximity based routing. The service broker is having additional responsibility of scale up the deployment of applications based on the load it is suffering from. This problem is solved by increasing or decreasing the number of virtual machines allocated in the data center. Figure 8 shows the calculation of response time using reconfigure dynamically with load policy in round robin algorithm whereas Figure 11 shows the same with weighted round robin policy which generates better results.

On the basis of this the results shown are as follows:

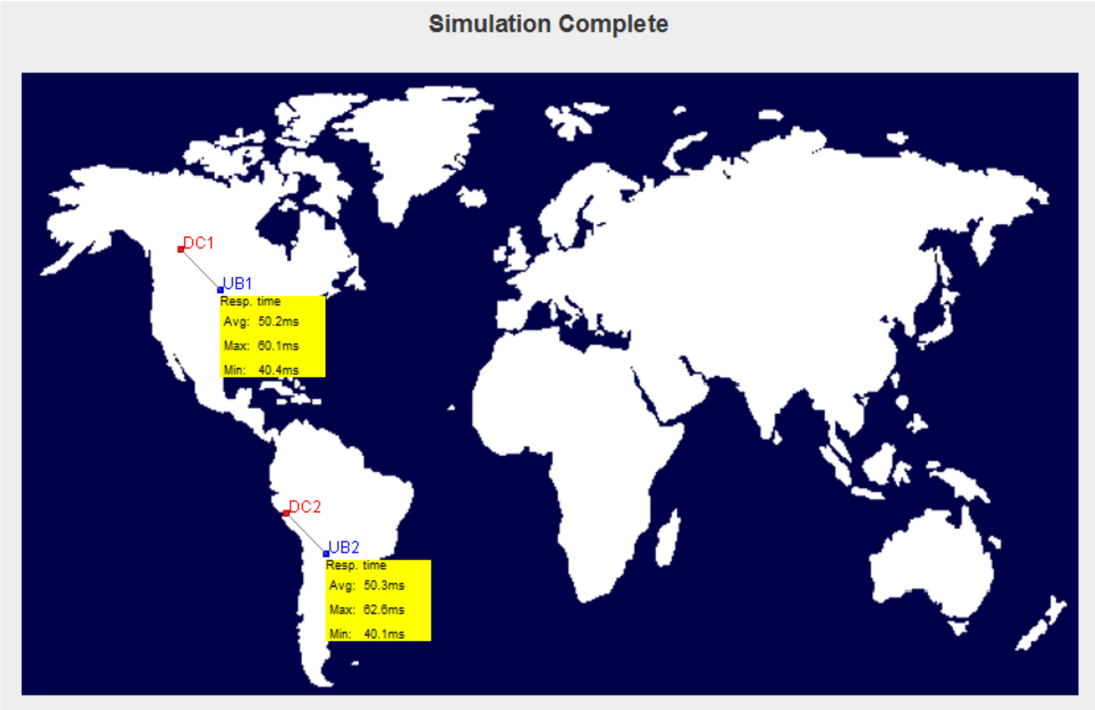

Figure 6: *Calculation of response time using closest data center policy via round robin algorithm*

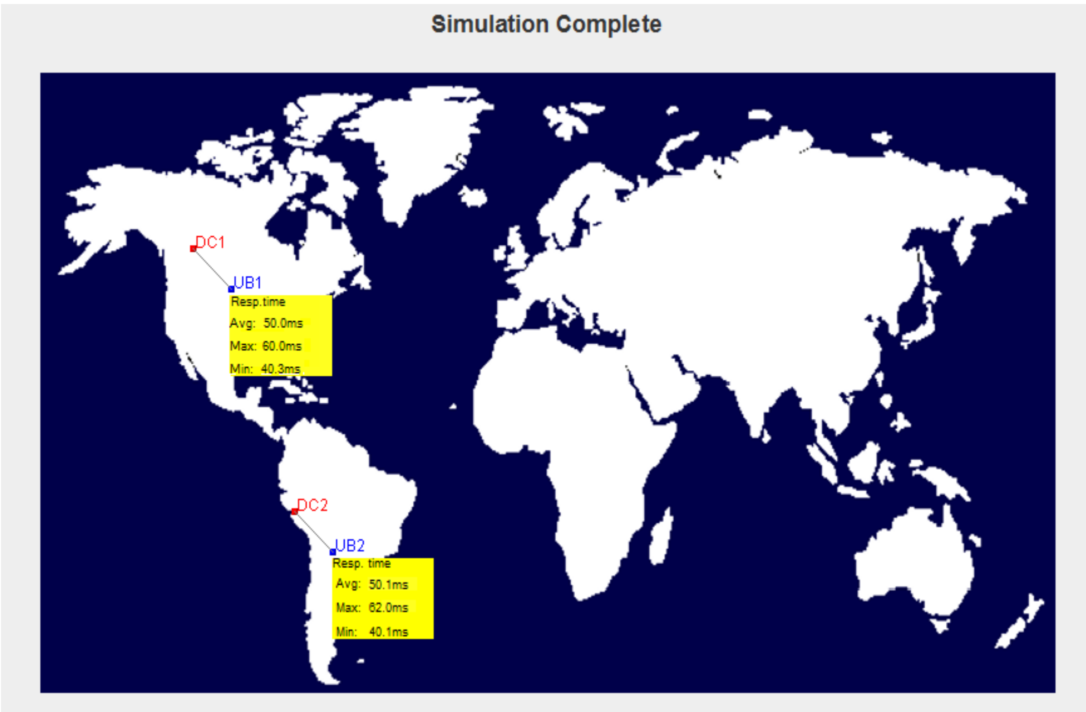

Figure 7: *Calculation of response time using optimal response time policy via round robin algorithm*

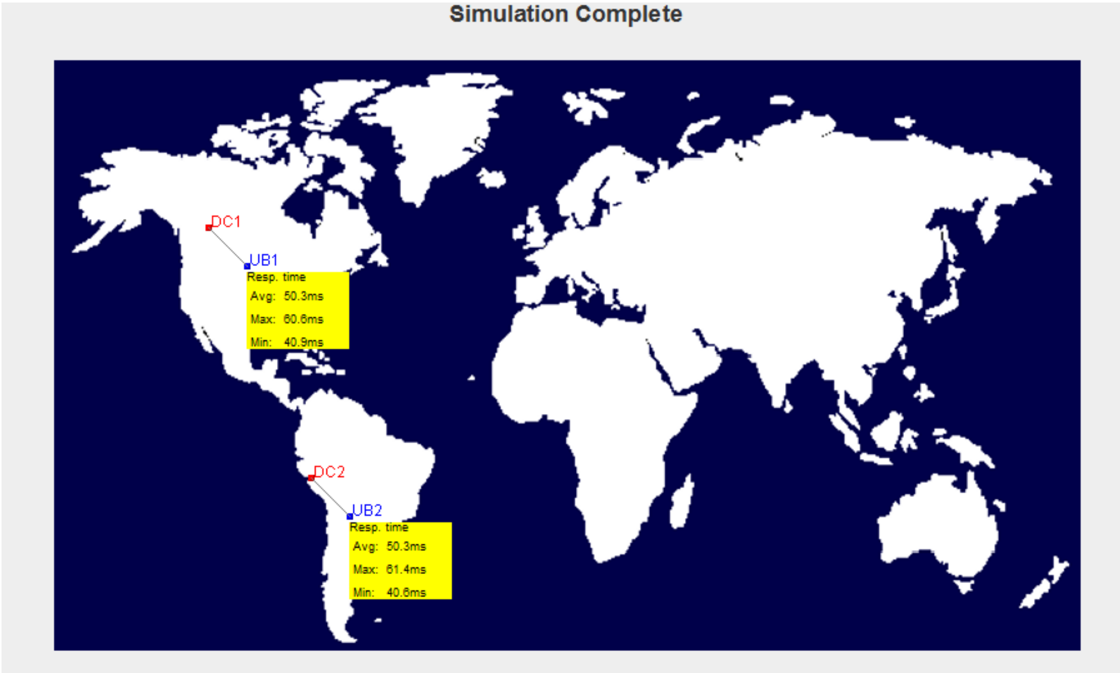

Figure 8: *Calculation of response time using reconfigure dynamically with load time policy via round robin algorithm*

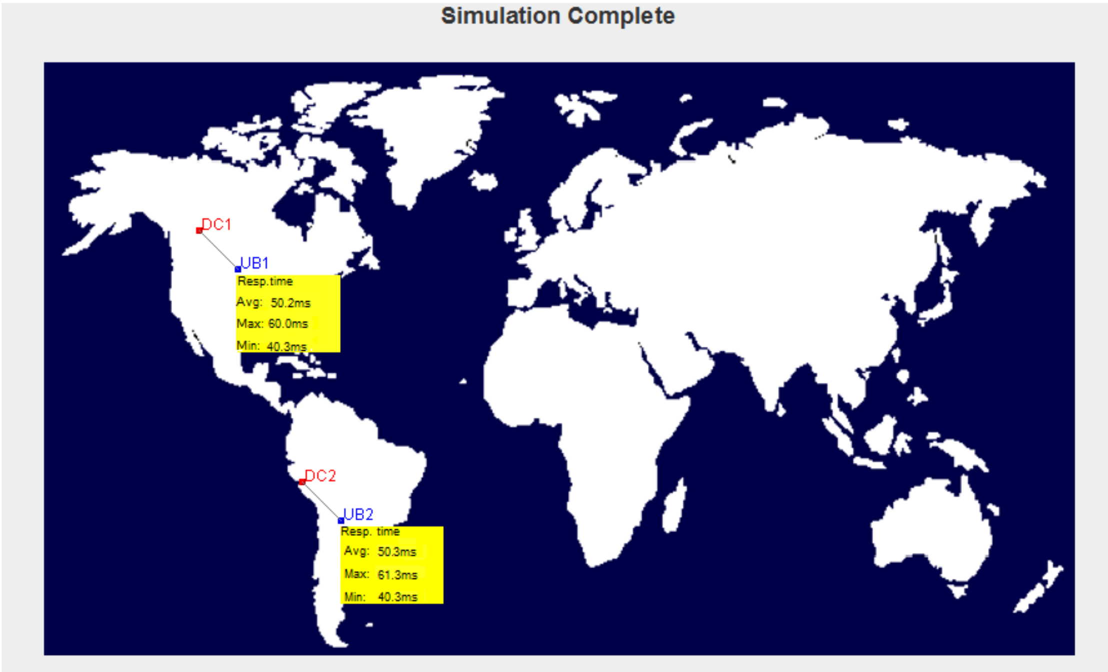

Figure 9: *Calculation of response time using closest data center policy via weighted round robin algorithm*

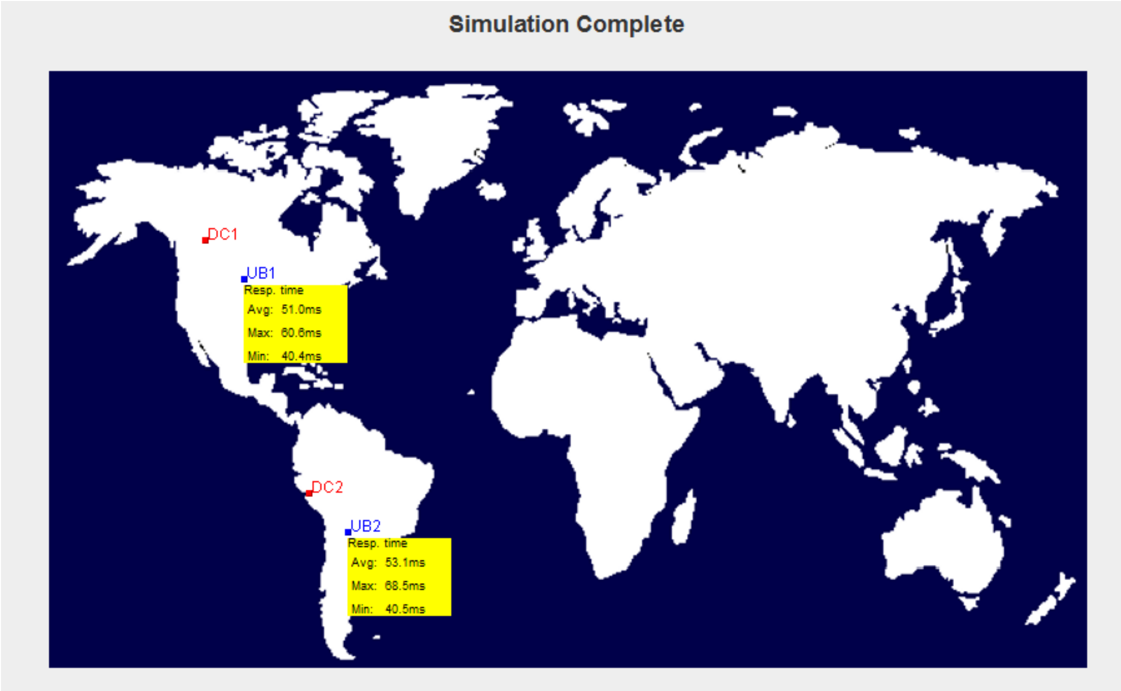

Figure 10: *Calculation of response time using optimize response time policy via round robin algorithm*

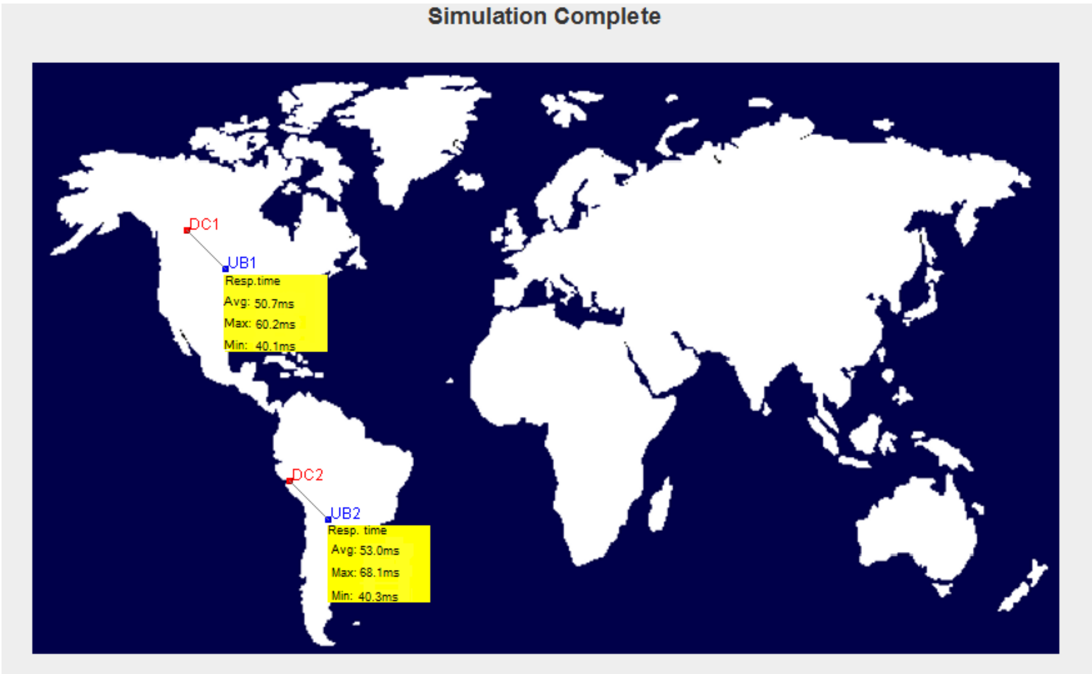

Figure 11: *Calculation of response time using reconfigure dynamically with load policy via weighted round robin algorithm*

6.5. Response Time on the Basis of Regions

In the CloudAnalyst the world is divided in to 6 regions which consist 6 main continents in the world. The other main entities like user bases and data centers belong to following regions. This geographical consortium is used to preserve a level of realistic ease for the large scaled simulation being attempted in the CloudAnalyst. Figure 12, Figure 13, Figure 14, show the comparison between two user bases of RR and WRR algorithm at average, maximum and minimum scale using ORT, CDC, and RDWL policies simultaneously. Approximate users according to CloudAnalyst per region.

***Table 1**: Approximate user according to CloudAnalyst per region*

Region	CloudAnalyst region Id	Users
North America	0	80 million
South America	1	20 million
Europe	2	60 million
Asia	3	27 million
Africa	4	5 million
Oceania	5	8 million

6.5.1. Results

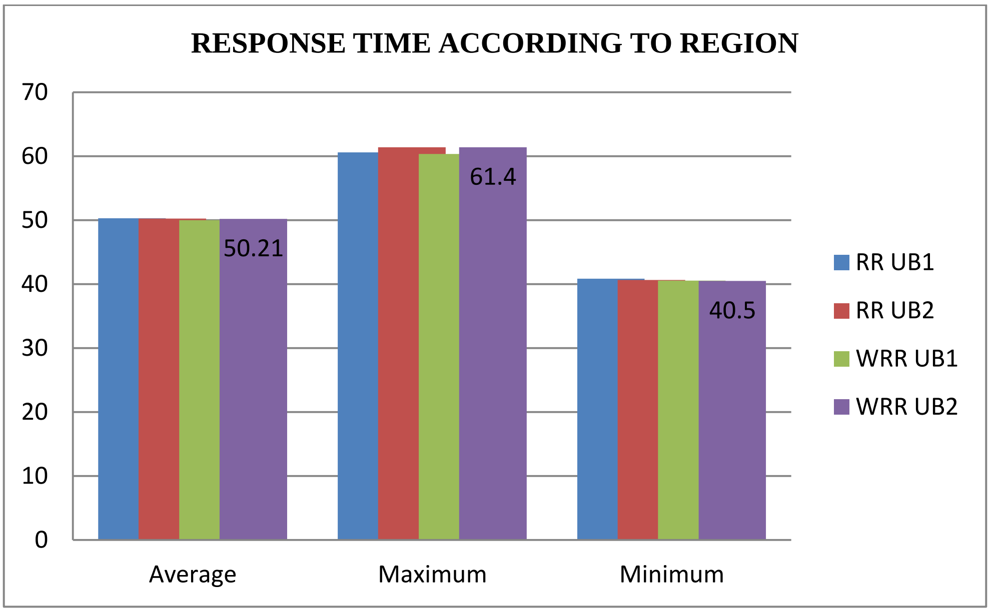

***Figure 12:** Optimise response time*

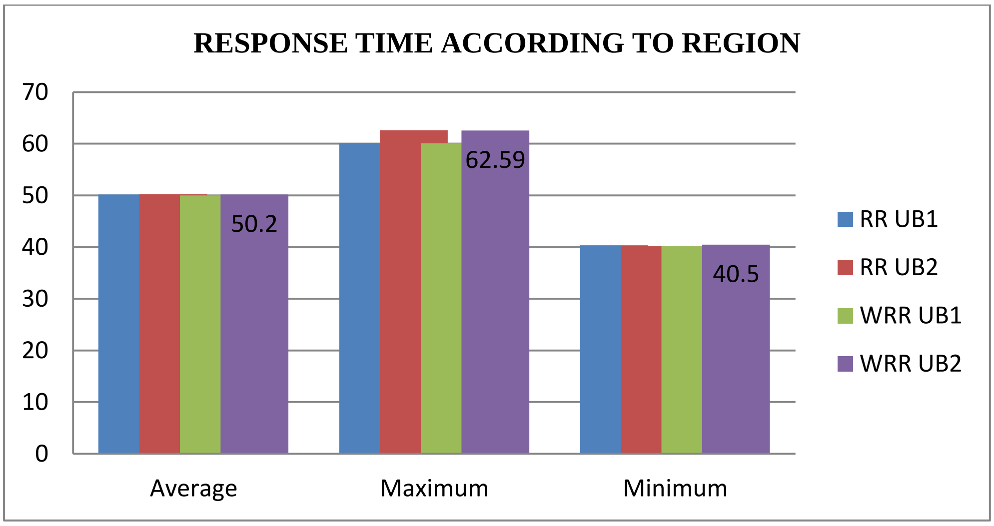

***Figure 13:** Closest data center*

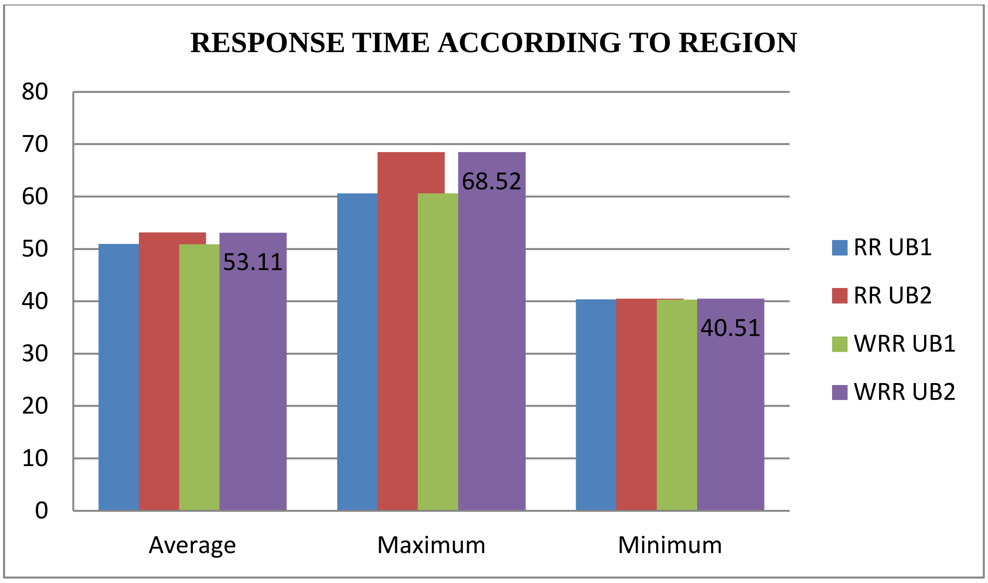

Figure 14: *Reconfigure dynamically with load*

6.6. Data Center Processing Time

Once the requests are received at the Data Center Controller it may again sub divided the requests in a single Internet Cloudlet in to multiple sub Internet Cloudlets based on the 'DC Request Grouping Factor'. Each of these sub cloudlets are then assigned to virtual machines by the load balancer and the new request is completed only when all the sub cloudlets are processed and returned to the controller. But this total duration is the time for processing all the requests in the internet cloudlet. If the data center controller waits till this point to send back the response to the user base then the final response recorded by the user base is the total processing duration plus the transmission delay for a single request. Therefore the data center controller is designed to send back the response to the original request on the receipt of the first response sub cloudlet. Figure 15, Figure 16, Figure 17, show the data center processing time of RR and WRR policies at average, maximum or minimum scale using CDC, ORT and RDWL policies simultaneously.

$$T_{DCP} = T_P + T_{Delay} \ldots\ldots\ldots\ldots \quad (2)$$

Where T_{DCP} is data transfer processing time, Tp is processing time, T_{Delay} is the delay time.

$$T_{Delay} = T_{Latency} + T_{Transfer} \cdots \quad (3)$$

Where $T_{Latency}$ is the network latency and $T_{Transfer}$ is the time taken to transfer the size of data of a single request (D) from source location to destination. $T_{Latency}$ is taken from the latency matrix (after applying Poisson distribution on it for distributing it) held in the Internet Characteristics.

$$T_{Transfer} = D/B_{W(Per\ user)} \ldots\ldots\ldots (4)$$
$$B_{W(Per\ user)} = B_{W(Total)}/N_r \ldots\ldots\ldots (5)$$

Where $B_{W(Total)}$ is the total available bandwidth (held in the Internet Characteristics) and N_r is the number of user requests currently in transmission. The Internet Characteristics also keeps track of the number of user requests in-flight between two regions for the value of N_r.

Figure shows the graphs of data center processing time with different cloud server broker policies:-

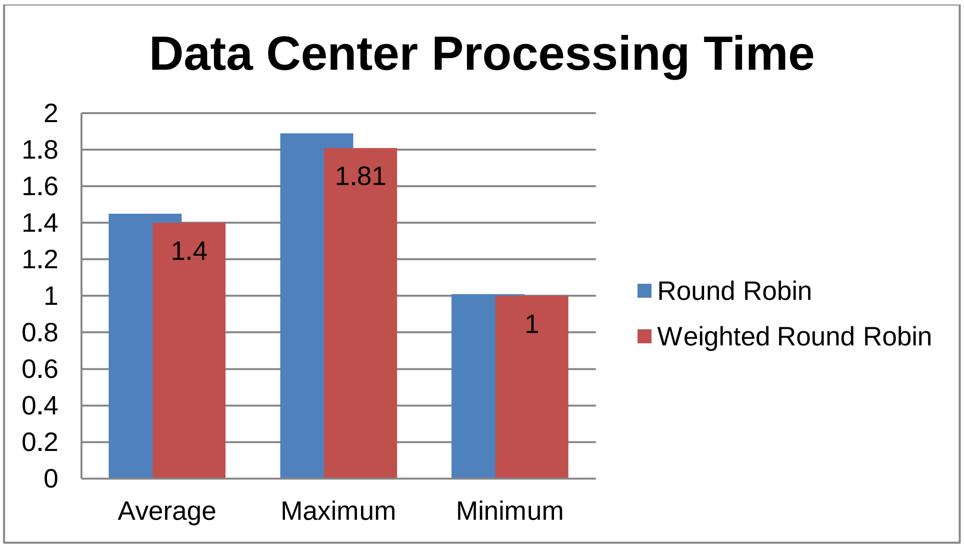

Figure 15: *Closest data center technique*

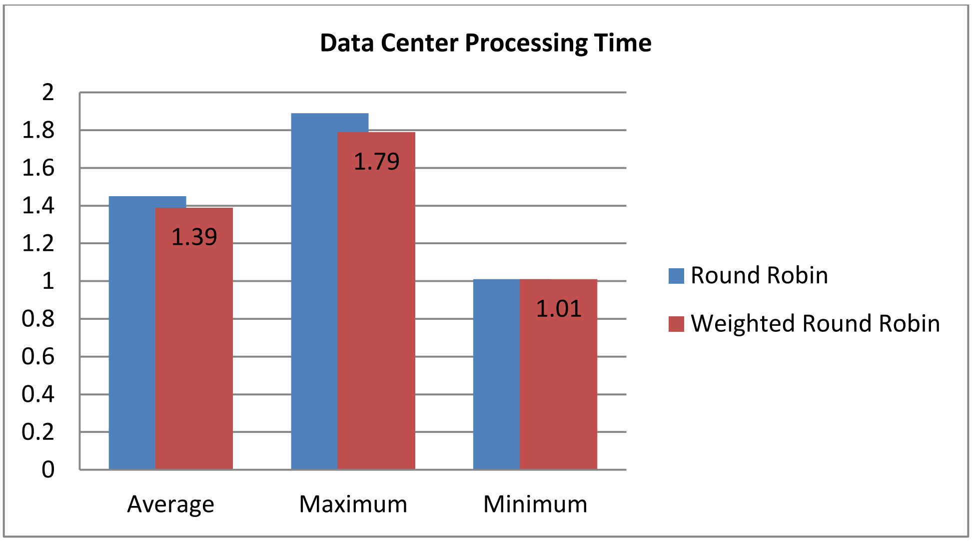

Figure 16: *Optimise response time technique*

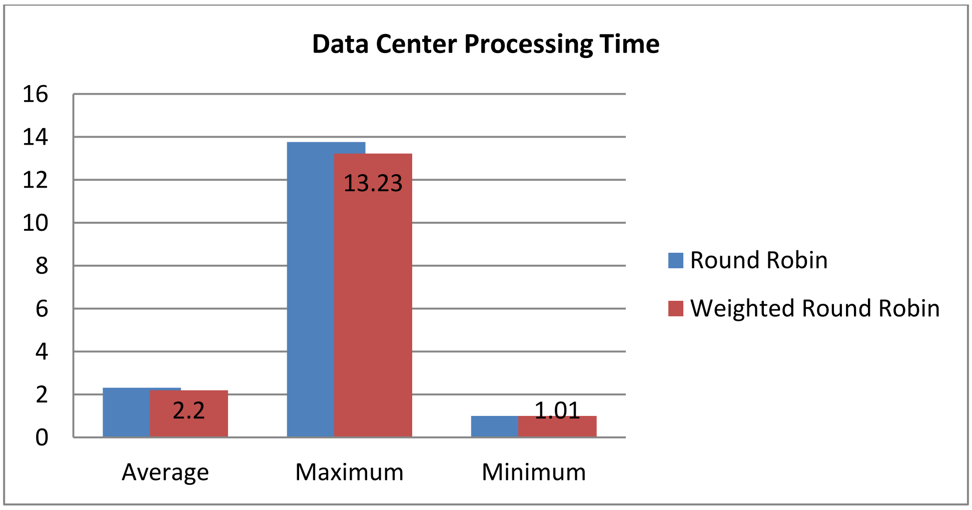

Figure 17: *Reconfigure dynamically with load technique*

6.7. Data Center Request Servicing Time

Data center request servicing time can be defined as the time taken by the server to fulfill the request of the user. Figure 18, Figure 19, Figure 20, show the data center request servicing time of RR and WRR policies at average, maximum or minimum scale using CDC, ORT and RDWL policies simultaneously.

$S = \frac{B}{C}$…………….(6)

Where S is the service request time, B is the busy time of the server and C is the completion time of the user's task.

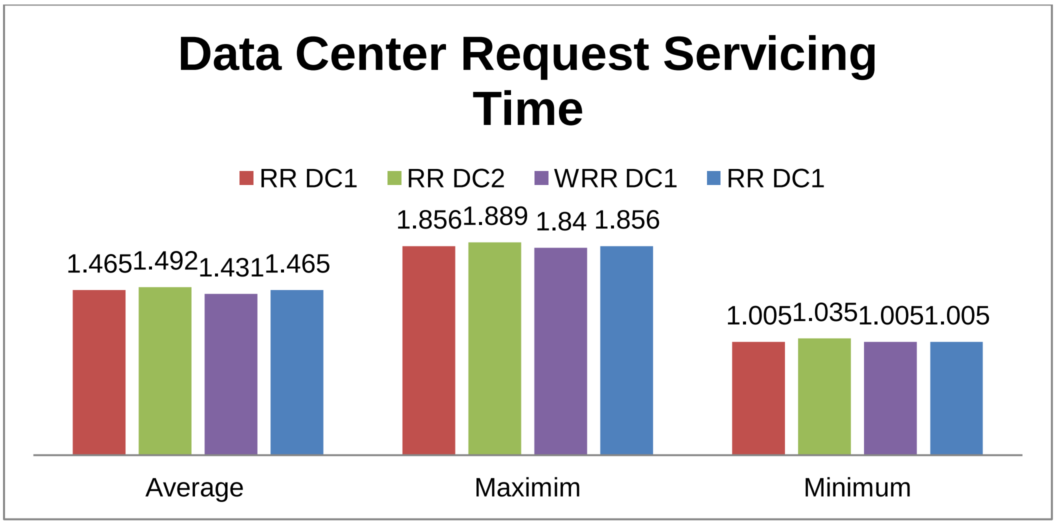

Figure 18: *Closest data center*

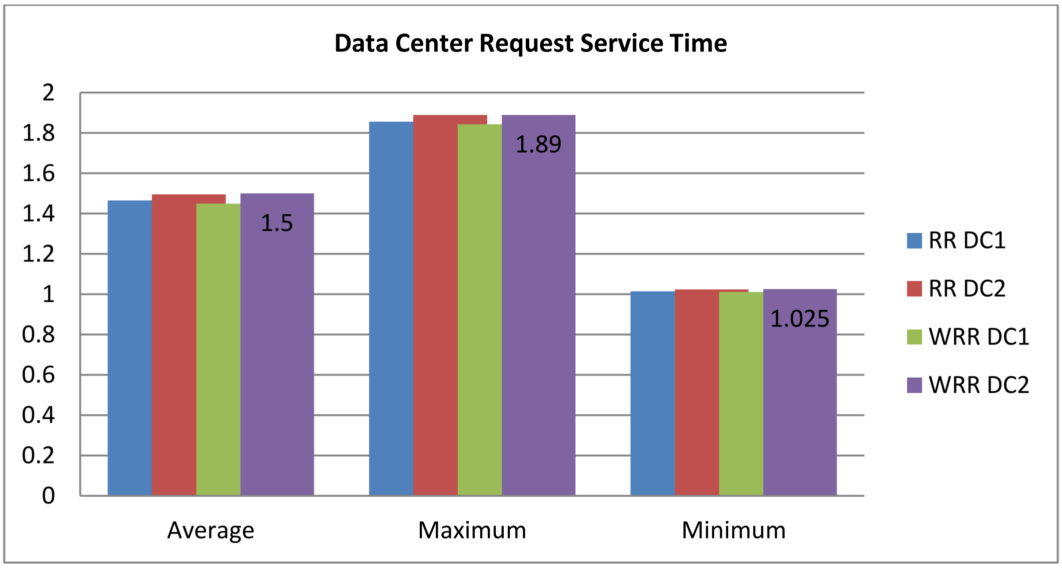

Figure 19: *Optimise response time*

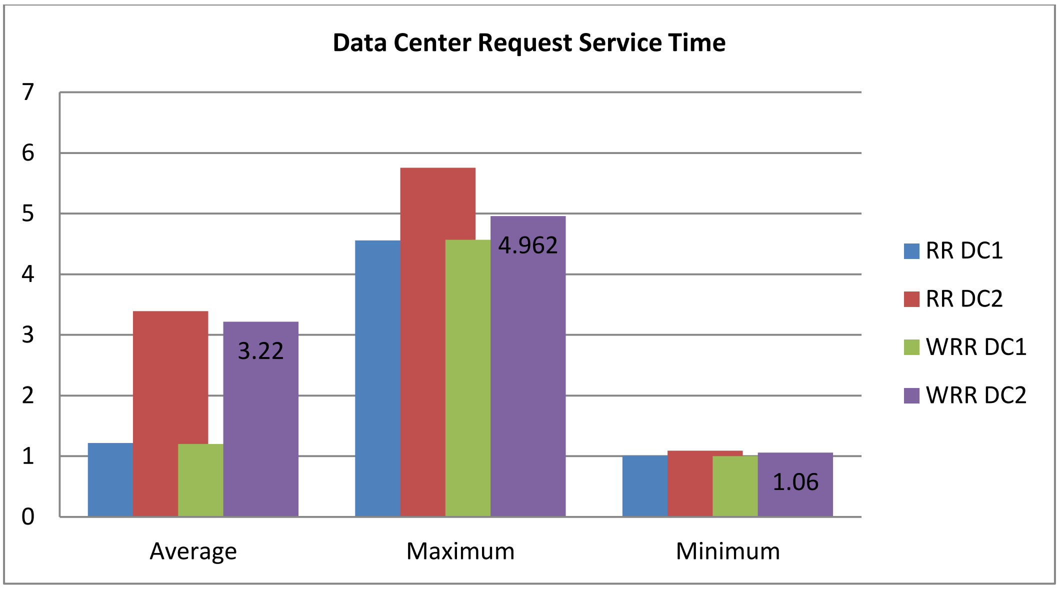

Figure 20: *Reconfigure dynamically with load*

7. Conclusion

At last it is concluded that the adoption of cloud is good from the several aspects. It's good to take resources on rental bases rather to buy it and best thing is that one has to pay only for the amount of time for which he used such resources. One more technology of cloud is focused here that is DBaaS (database-as-a-service). It is beneficial to store the data on cloud database because personal systems are not enough spacious to store such huge amount of data there. In this case cloud database is one of the options to store their data. There are less chances of data loss in cloud databases as comparison to the normal database or we can say that cloud databases are robust in the case of disaster or data loss.

Also the information about cloudsim based tool CloudAnalyst is incorporated here. The tool is used for performing simulation using various policies. It maps the world wide data and connects several data centers to user bases. It also performs the task of load sharing in the case of overloading. There are three algorithms used in the simulator for performing the task of load sharing round robin policy, throttled load balancer policy and the equally spread current execution policy fourth one is the proposed algorithm for performing the task of load balancing on cloud database that is the weighted round robin policy. If we comparing the result of algorithms then the response time of round robin policy is smaller amongst all. With the service broker policy of closest data center it produced better results. Although the experiments performed only on two datacenter or five virtual machines and the generated result having minor difference from the previous one. But these changes must be taken into account and with some improvements or future understandings it could produce better results.

References

[1] Buyya, R., Yeo, C.S., and Venugopal, S. Market-Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Utilities. Proc. of the 10th IEEE International Conference on High Performance Computing and Communications (HPCC 2008, IEEE CS Press, Los Alamitos, CA, USA), 25-27, Dalian, China, Sept. 2008.

[2] Buyya, R., Ranjan, R., and Calheiros, R.N. 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.

[3] Buyya, R. and Murshed, M. GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing. *Concurrency and Computation: Practice and Experience*. 2002. 14; 1175-1220.

[4] Buyya, R. and Murshed, M. GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid computing. *Concurrency and Computation: Practice and Experience.* 2002. 14 (13-15) 1175-1220.

[5] UOE (University of Edinburgh). Institute for Computing Systems Architecture. Simjava. 2009. http://www.dcs.ed.ac.uk/home/hase/simjava

[6] Bhathiya Wickremasinghe. CloudAnalyst: A CloudSim-based Tool for Modelling and Analysis of Large Scale Cloud Computing Environments. MEDC Project Report. Sep 2009.

[7] Pizzete, L. and Cabot, T., 2012: Database as a Service: A Marketplace Assessment. Retrieved 23rd November from http://www.mitre.org/work/tech_papers/2012/11_4727/cloud_database_service_dbaas.pdf.

[8] Sivashakthi, T. and Prabakaran, N. A Survey on Storage Techniques in Cloud Computing. *International Journal of Emerging Technology and Advanced Engineering*. 2013. 3 (12) 125-128.

[9] Nusrat Pasha, Amit Agarwal, and Ravi Rastogi. Round Robin Approach for VM Load Balancing Algorithm in Cloud Computing Environment. *International Journal of Advanced Research in Computer Science and Software Engineering.* 2014. 4 (5) 34-39.

[10] Waleed Al. Shehri. Cloud Database as a Service. *International Journal of Database Management Systems*. 2013. 5 (2) 1-12.

[11] Gill Sukhjinder Singh and Thapar Vivek. Implementation of a Hybrid Load Balancing Algorithm for Cloud Computing. *International Journals of Advanced Technology in Engineering and Science.* 2015. 10 (2) 73-81.

[12] Wickremasinghe Bhathiya, Rodrigo N. Calheiros, and Buyya Rajkumar. Cloud Analyst: A CloudSim Based Visual Modeler for Analyzing Cloud Computing Environments and Applications. Distributed Computing Project, CSSE Dept. University of Melbourne, 2009.

[13] Gupta Ruhi. Review on Existing Load Balancing Techniques of Cloud Computing. International *Journal of Advanced Research in Computer Science and Software Engineering*. 2014. 4 (2) 168-171.