International Journal of Grid and Distributed Computing
Vol. 9, No. 9 (2016), pp.11-24
http://dx.doi.org/10.14257/ijgdc.2016.9.9.02

Analysis of Load Balancing Algorithms using Cloud Analyst

Simar Preet Singh[1], Anju Sharma[2] and Rajesh Kumar[3]

[1,2,3]*Thapar University, Patiala-147004, India*
[1]*er.simarpreetsingh@gmail.com*
[2]*anjusharma@thapar.edu*
[3]*rakumar@thapar.ed*

Abstract

This paper discusses the Cloudanalyst tool. Cloudanalyst tool is used to determine the better load balancing algorithm from various scheduling and load balancing techniques e.g. round robin algorithm. This learning will help valued understanding to design infrastructure services of the Cloud. Different areas like coordination between one data center and other data center, algorithms of load balancing as well as other value-added services are also kept in mind, that are possible like service broker policies, which synchronize efficiently among data centers and enhance cost and throughput which is given to the owners.

Keywords*: Cloud analyst and its features, Data Center(DC), Virtual machines(VM), User bases(UB), load balancing algorithms, service broker algorithms.*

1. Introduction

Cloud computing tends to be the emerging computing model where the advancement of cloud technology has unlocked many new opportunities to developers that use Internet applications recently. Earlier deployment of application and hosting of the application was one of the main concern and was given the priority while designing an internet application but now with introduction of the cloud it is possible to solve the problem of deploying and hosting the applications more flexibly and economically using the significant infrastructure services which are provided through cloud service providers. Cloud computing is the pay as you use model and whenever the service is required, with few clicks it is accessible.

The US NIST proposed a definition for cloud computing as a technology that enables pervasive, i.e. present everywhere, on-demand and convenient internet fetching to shared group of configurable computing devices like servers, networks, services, softwares and storage devices which may be accessible fast and are brought in market with slight service provider interaction or minimum management effort involved [5]. There are existing many tool-kits which can also be used for simulation environment and can help in studying the performance of scalable applications which are available on the Internet [7].

Cloudanalyst seems to be easy as it has a graphical user interface with which it seems like easy to use tool and also seems to have a level of visualisation capability which is even better than just a tool-kit. Cloudanalyst separates simulation set up environment exercise and supports the modeller to focus on the parameters used for simulation purposes rather than the programming technicalities only. It also supports a modeller to perform simulations continually by modifying the parameters easily, quickly and in very less time [2]. The graphical user interface simulation output results of cloudanalyst enables the results to be examined more straight forwardly with more proficiently and also quickly helping in stressing out any problems dealing with the accuracy and performance of simulation.

ISSN: 2005-4262 IJGDC

1.1 Features of Cloud Analyst Simulator

Cloudanalyst provides number of features being a simulator. Some of the features of the simulator are as follows:

1.1.1 Simple to Use: Cloudanalyst is easy in setting up as it comes with the java package and we need to simply double-click on the icon. In this, simulation experiment execution is the main key feature of cloudanalyst simulation tool.

1.1.2 GUI based Output: GUI based output which comprises of tables (rows and columns), graphs and charts are highly desired to review a number of results which are obtained at the time of Cloud Analyst simulation. These such GUI based presentation helps in understanding and identifying the important outlines of the parameters and also helps in their comparison.

1.1.3 Ability to Repeat: Cloudanalyst has the ability to repeat the experiments. This is a very significant requirement of any simulator. With cloudanalyst simulator, if one experiment having some parameters, on simulation, produces some results then these results will be same each time the same simulation is executed with same parameters in the same experiment. Without this, the simulation just seems to be a random sequence of events and not the controlled experiment.

1.1.4 Ability to Save the Results: Cloudanalyst also has the option to save the results. This is helpful as we can save the experiment (along with the set of all input parameters and values taken during simulation) as a file. This file can then be saved on the system (PC) or the same can be taken into flash drives or pen drives to some other computers at different locations.

2. Use of Cloudsim Toolkit

Cloud Analyst runs on cloudsim. The cloudsim toolkit includes the activities that are taking place in the Data Center mainly and that too in detail [7]. These activities include:

- In terms of physical machines, it simulates the data center's hardware definition, which comprises of memory, storage devices, bandwidth and processors.
- It also simulates the creation, deletion and virtual machine specification.
- It also provides virtual machines management, the hardware components allocation to perform the VM's operations on the basis of diverse procedures available (like time-based policy, space-shared policy etc).

2.1 Introduction of New Extensions

Cloudsim toolkit provides data center operations. In addition to these, cloudsim toolkit also provides following extra functionality, which is the main requirement for the cloudanalyst to execute and thus helped to run the cloudanalyst on the top of cloudsim [4].

2.1.1 Autonomous Entities: Entities that are autonomous are necessary to act as a generator of traffic and they must be configurable in their behavior.

2.1.2 Network delays and Bandwidth restrictions: The data which is transmitting on the Internet must be accurately modelled which seems to be a realistic one, also including delays in network and restrictions on bandwidth so as simulation can be done correctly.

2.1.3 Simulation defined by Time Period: The package of cloudsim that works for operating on some default actions. (such as acceptance of n-number of cloudlets). There is a requirement to change the simulation to time-frame restricted implementation that

considers the events that are unceasingly produced unless a default time frame is achieved via users.

2.1.4 Service Broker: Cloudsim is embedded with intellection of data center brokers that enforces a double action in virtual machine supervision of different data centers plus the moving traffic to most suited data center's. The case is different in cloud analyst where both these roles are divided and drafted to multiple entities. The data center broker is supplemented by the data center controller which is obligated for VM management that resides in one data center that in turn embeds the load balancing of virtual machines under the same data center.

2.1.5 Graphical User Interface: Cloudanalyst is inflected with a widespread GUI that constructs the simulation process in an elaborated manner [8]. The GUI is used to configure and implement simulation processes in a simple and repeatable way that gives the user an advantage of reliving him from focusing the execution and certainty of simulation code that improves the whole process.

2.1.6 Capacity to Save the Simulations and the Final Results Obtained: In addition to all the features, the cloudanalyst allows provides save option, using which the user can save all the simulation settings/configuration as an XML file for later use and the same can also be ported from one computer to another or from one location to another. The saved results can be exported back to resume the configurations done during the simulation. The results can also be exported into PDF format and then can get the hard copy print of them.

2.2 Main Components of Cloud Analyst Simulator

Main components of cloudanalyst simulator are as follows [4]:

2.2.1 Regions: Cloudanalyst simulator split the whole world into six regions. These regions coincide with six main continents in the World. Data centers and user bases, which are other main entities, related to some of these areas. This ecologically based categorization is very useful for maintaining the level of real-based simple and easy model for a very large duration simulation, which involves large parameters, are all performed in the Cloudanalyst and the results are obtained.

2.2.2 Internet Settings: Internet in the cloudanalyst simulator is the idea for the actual real world Internet. This implements all those features which are important only to that particular simulation. Cloudanalyst also manages to create the replicas of the Internet traffic, which seems to be routing around all over the sphere through hosting appropriate amount of data transmission delays and the transfer latency. The user can also configure the transmission latency and all the existing bandwidth among six areas (the whole world is divided into regions).

2.2.3 Service Broker: Service Broker manages all traffic that is moving between the data center's and the user bases. The decision about servicing of the data center from each user base is done by the Service Broker. Cloudanalyst equipment presently provides three types of service brokers and all these three types of service brokers implement different routing policies [10]. The three service brokers are: closest data center, optimize response time and dynamically reconfigured.

2.2.4 User Bases: The collection of users, considered to be as one individual unit in the cloudanalyst, which are involved in the simulation, are said to be the User Bases. The

main responsibility of user bases is generating traffic for cloudanalyst simulation, with which the simulation considers being in the real time [6]. Thousands of users can be represented as a single User Base but all these users are configured as a single unit i.e. a single user base. Traffic generated in these user bases will be the simultaneous bursts depending on the capacity of the respective user base. User Base can be chosen by the modeller in order to signify each user since we have to increase the efficiency of the simulation, generally, the user bases are used to be considered as they represent a large number of users.

2.2.5 Data Center Controller: This Controller proves to be very indispensable part in cloudanalyst. One particular cloudsim is mapped to each data center controller. Data center controller manages the activities performed by the data center. These activities are like the creation of VM's etc. Data center controller also routes user requests that are received from respective user bases with the help of the Internet to the virtual machines. This can also be observed as the facade which is used by the cloudanalyst for accessing and functioning the main part of cloudsim toolkit.

2.2.6 VM Load Balancer: The data center utilizes the virtual machine load balancer. The data center controller make use of virtual machine load balancer to govern the particular virtual machine allocated for processing of the upcoming cloudlet. Presently cloudanalyst has three VmLoadBalancers which implements three different load balancing procedures and their policies can be further selected by the modeller according to the requirement. The three different VM load balancer are: active monitoring, round-robin and throttled load balancer. A brief description of these three VM load balancer are as follows:

a. Active Monitoring Load Balancer: This Load Balancer stabilities the total available jobs in the total existing VM's in such a manner that this load balancer tries to balance total existing active jobs that are running on any virtual machine at any given time.
a. Round-Robin Load Balancer: This load Balancer uses the concept of the simple round-robin algorithm for the allocation of the virtual machines. Round Robin algorithm is the algorithm which is dependent on time and after the regular interval of time, allocation of the virtual machine is changed to the next available virtual machine.
c. Throttled Load Balancer: In Throttled Load Balancer, at any given time, only the certain quantity of the internet Cloudlets will be distributed to the particular virtual machine. The requests will be queued up if the virtual machine receives more number of request groups than it can handle, and whenever any virtual machine becomes available at any time, then that virtual machine gets all the queued up requests for processing.

3. Algorithms Used

We have used various algorithms to perform simulation in CloudAnalyst and obtain the results of the paper [3]. The data centers use the VM Load Balancer and by using it, Data Center balances load requests between all the available virtual machines [1]. The various algorithms that are used are as follows [4]:

3.1 Algorithms: The various algorithms used in this paper are as follows:

3.1.1 Throttled Load Balancing Algorithm: This Load Balancing Algorithm is as follows:

1. The index table of all the virtual machines is maintained by ThrottledVM LoadBalancer. This also maintains the state of each virtual machine i.e. whether the

virtual machine is busy or available. Initially, at the start of the algorithm, all the virtual machines have been present

2. Then, DataCenterController gets the fresh task.
3. DataCenterController, then on receiving the call, contacts Throttled Virtual Machine LoadBalancer to do the next allocation of the virtual machine.
4. Then Throttled virtual machine LoadBalancer deconstructs virtual machine allotment table starting from top to bottom till the current accessible virtual machine is located. The table must be scanned entirely.

If available virtual machine is found, then:

a. VM id is returned by the Throttled VM Load Balancer to the DataCenterController.
b. The DataCenterController then transfers the call to a respective virtual machine which has been identified by that particular virtual machine id.
c. Then DataCenterController gives the notification to the Throttle VM load balancer about the allotment about the new virtual machine id.
d. On receiving the call from the Data Center Controller, ThrottledVM LoadBalancer then upgrades the virtual machines allotment table consequently.
When available virtual machine is not found then:
a. -1 is returned by the ThrottledVM LoadBalancer.
b. Then the request is put into the queue by the DataCenterController.
c. When all the processing request is completed by the virtual machine and the response has also been received by the DataCenterController, ThrottledVM Load Balancer gets a notification from Data center controller to perform the de-allotment of the respective virtual machine.
d. Now when the virtual machine is de-allocated, then the DataCenterController examines the awaiting call queue. When some waiting calls in the pending queue exist, the processing of the particular call starts from 3rd step onwards.
5. Proceed from step 2.

3.1.2 Active Monitoring Load Balancing Technique: This Balancing technique tries to have an equal amount of work on all the available virtual machines. This algorithm is nearly the same to Throttled Load Balancing technique in use. The steps involved in Active Monitoring Load Balancing algorithm are as follows:

1. Index table of all the virtual machines gets updated by the ActiveVM Load Balancer and this also keeps the record of the amount of calls being presently allotted to each VM. Initially, every VM has 0 allocations.
2. Data Center Contoller requests for the allocation of a new virtual machine. When this request comes, it goes through the complete virtual machine allocation table and then identifies which of the available virtual machines has the least load. If the least load of two or more virtual machines are same, then the first identified virtual machine will be allocated.
3. Then, after identification, the virtual machine id is returned back from the ActiveVM LoadBalancer to the DataCenterController.
4. After this, DataCenterController transfers the call to the selected VM and the corresponding ID recognizes the VM.
5. Also, the DataCenterController notifies about the new allocation of the virtual machine to the ActiveVM LoadBalancer.
6. Then the allocation table is updated by the ActiveVM LoadBalancer and also the allocation count for that particular virtual machine is increased.
7. When the virtual machine completes processing the call then the DataCenterController gets the corresponding cloudlet. When this happens, them it also notifies the ActiveVM Load Balancer of VM that de-allocates the virtual machine.

8. Then finally the allocation table is updated by ActiveVM LoadBalancer that does it through lowering the allocation count for that particular virtual machine by one.
9. Continue from step 2.

3.1.3 Round Robin Load Balancing Algorithm: This load balancing technique uses the concept of round robin. Time quantum is used in this method. This algorithm proceeds in the following way:

Round_Robin_Load_Balancing_Algorithm ()
{
Firstly, make the status of all available virtual machine to be present in the virtual machine state list. After doing this, start the hash map without any values. Then perform the while loop as below:
While(condition of this while loop is: Data Center Controller receives the new request)
do
{
requests are queued by the Data Center Controller;
the request is being removed by the Data Centre Controller from the beginning of the queue; After this, the decision making is performed:
if(do the hash map contain any of the entry of a virtual machine which corresponds to the user base that is being currently requested AND (&&) compare the virtual machine status (whether its available or not) i.e. allocation status of a VM == Available)
{
VM is allocated to that particular user call
}
else
{
Using the Round Robin Algorithm, virtual machine is allocated to the user base request;
Then entry of the user base is updated and the virtual machine entry is also updated in the hash map as well as VM state listing;
}
}
}

3.2 Service Broker Algorithms: In Cloud Analyst, there are various Service Broker Algorithms. These are as follows:

3.2.1 Performance Optimized Routing: BestResponseTimeServiceBroker implemented the Performance Optimized Routing rule. This rule extends the ServiceProximityServiceBroker. The main steps covered in this policy are:

1. Index of all the Data Centers available is maintained by the BestResponseTimeServiceBroker.
2. The Internet receives the user base's message and when it receives then it requests to allot it with the desired position of the DataCenterController. The query is done by BestResponseTimeServiceBroker.
3. Considering the latency, applying the technique ServiceProximityServiceBroker, the nearest data center is identified by the BestResponseTimeServiceBroker.
4. Then listing contains every data center which is gone through by the BestResponseTimeServiceBroker and then with this iteration of the entire data centers, the present response time of every individual data center is estimated through:

a. Final noted processing time is queered from the network properties.
b. The processing time of a particular data center is reset to zero if the last recorded processing time is recorded that too before a predefined threshold. From this, it seems like

the data center must have stayed idle for the duration of slightly equal to the amount of the threshold time.
c. The value obtained from the above steps is added further to the network delay obtained from the Internet Characteristics.
5. Here the decision making is performed. Closest data center is selected by the Best Response Time Service Broker when smallest calculated response time being given to data center which is the nearest.
Otherwise, if not then sometimes the nearest data center is selected or the data center having smallest response time bearing a half and half probability is picked by theBestResponseTimeServiceBroker.

3.2.2 Service Proximity-Based Routing: Service Proximity-Based Routing gives the easiest implementation of the Service Broker algorithm. The steps are as follows:

1. The region of the index table highlights the index table for every data center and ServiceProximityServiceBroker maintains them.
2. The Internet, on receiving the userbase's message, requests the ServiceProximityService Broker and using this query, it asks for the DataCenterController's destination.
3. The region of the sender's request is retrieved by the ServiceProximityServiceBroker. The queries for the areas proximity listing for that particular region is also retrieved by the Service Proximity Service Broker among network properties. With the mentioned listing, ordering of the left over areas is done, according to the smallest network latency, while network latency is calculated from the given region.
4. Firstly, the data center is situated at the biggest/nearby area in the proximity listing is picked by the ServiceProximityServiceBroker. If the number of data center increases from one to more that resides in one area, then single data center will be chosen at random.

3.2.3 Dynamic Service Broker Algorithm: The DynamicServiceBroker algorithm is the algorithm which is obtained by either extending the ServiceProximityServiceBroker algorithm or extending the finest ResponseTimeServiceBroker algorithm. The steps carried out are as follows:

1. A list of finest response time taken till that time for every particular Data Center and another list having all the data centers, these two lists are maintained by the DynamicServiceBroker.
2. When the message is received by the Internet from the user base, the user base makes the query to the DynamicServiceBroker to get a desired position of DataCenterController.
3. DynamicServiceBroker algorithm makes use of both the algorithms i.e. BestResponseTimeServiceBroker algorithm and ServiceProximity Service Broker algorithm for identifying the destination.
4. Finally, when present response time comes out to be superior in comparison to initially recorded response time, then the DynamicServiceBroker updates the best-recorded response time records.

4. Simulation Scenario

The cloudanalyst tool is used to analyze the various load balancing algorithms. To implement the various algorithms, we have simulated the environment by taking 6 user bases and 6 data centers, having 6 VM's in each data center. Each simulation is performed for 60 hours. We have considered average peak users as 10000 and average off-peak users to be 100 in each user base. Service broker policy used for simulation is nearest data center.

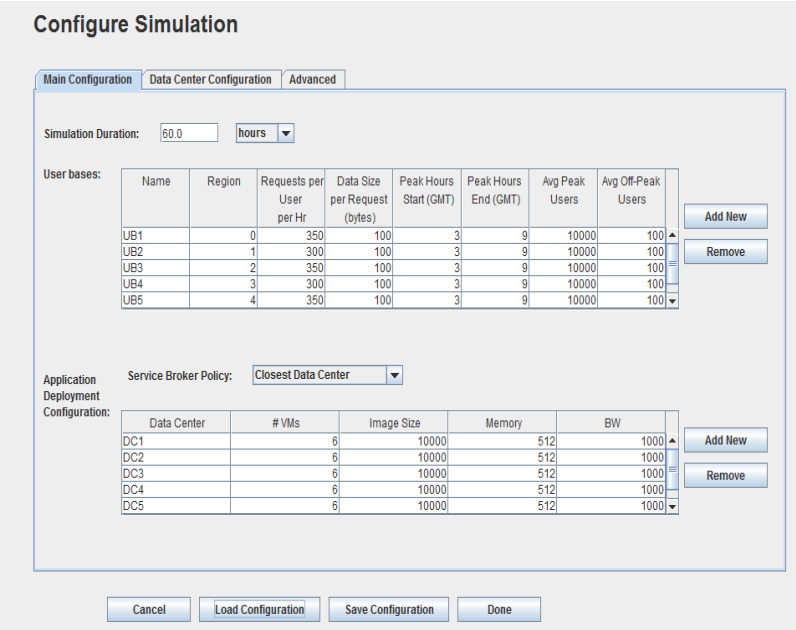

Figure 1. Main Configuration of Cloud Analyst

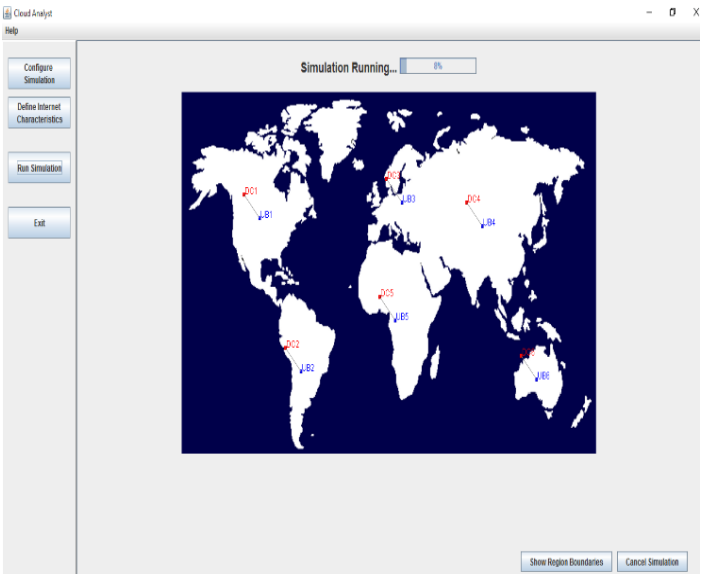

Figure 2. Simulation Running using Cloud Analyst

The various algorithms are compared by taking different scenarios as follows:

4.1: Scenario A, when Round Robin Load Balancing Algorithm is Applied: In this scenario, round robin algorithm is applied for finding the load on every virtual machine and for equalizing the load of each VM and the result is depicted in the Table1.

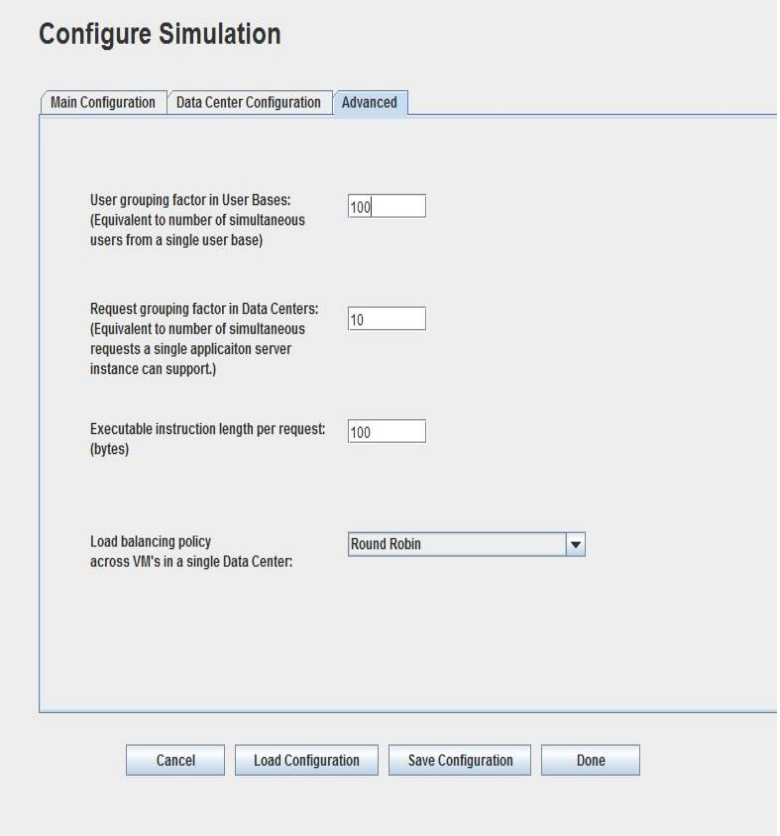

Figure 3. Advanced Configuration using Round Robin Load Balancing Policy

4.2. Scenario B, when Equally Spread Current Execution Load Balancing Algorithm is Used: The scenario depicts that instead of round robin algorithm, equally spread current execution load balancing algorithm is used for finding the load on each VM as well as to balance the load of each VM and the result is depicted in Table2.

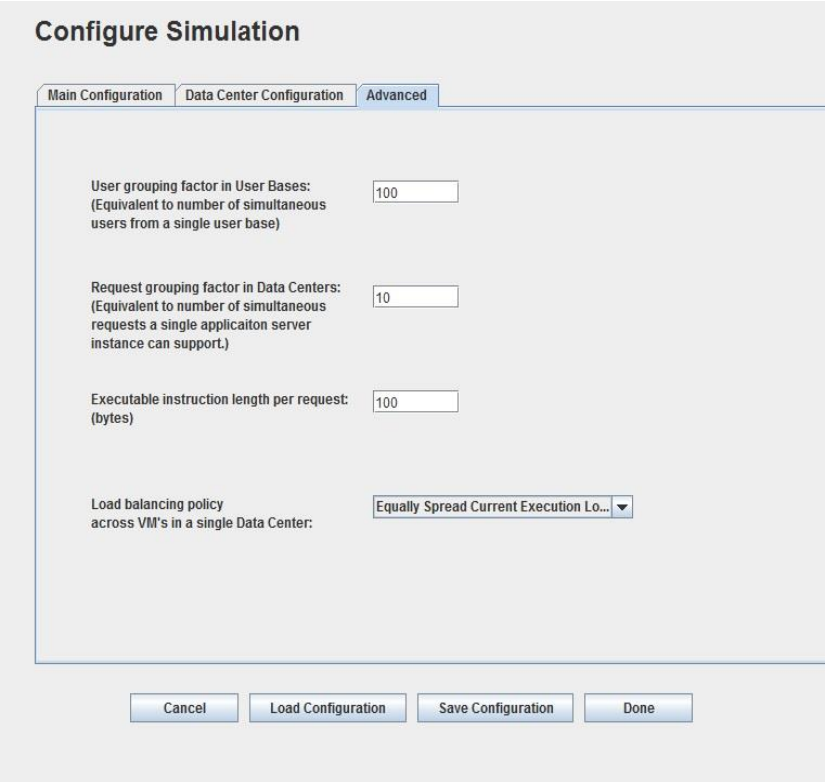

Figure 4. Advanced Configuration using ESCE Policy

4.3: Scenario C, when Throttled Load Balancing Algorithm is used: The given scenario presents throttled load balancing algorithm which is used for finding the load on each VM and to equalize the load of each VM and result is depicted in Table3.

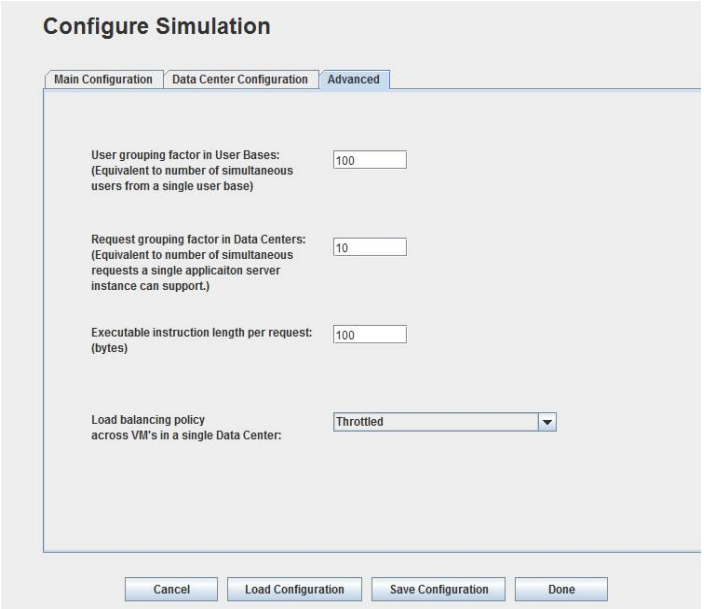

Figure 5. Advanced Configuration using Throttled Balancing Policy

5. Results

The results obtained after simulation of the various scenarios considering for different algorithms are shown in Table1, Table2 and Table3.

Scenario A, when round robin load balancing algorithm is used, the results obtained are shown in Table1:

Table 1. Results of Round Robin Load Balancing Algorithm

Parameter	Average (ms)	Min. (ms)	Max. (ms)
Total Response Time	64.04	34.5	9818.03
Data Center Processing Time	13.7	0.02	9768.77

The results obtained for the cost parameter after simulation of the senerio A is 1444.53$.

Scenario B, when equally spread current execution load balancing algorithm is used, the results obtained are shown in Table2:

Table 2. Results of Equally Spread Current Execution(ESCE) Algorithm

Parameter	Average (ms)	Min. (ms)	Max. (ms)
Total Response Time	64.38	34.5	9818.03
Data Center Processing Time	14.04	0.02	9768.77

The results obtained for the cost parameter after simulation of the senerio B is 1444.53$.

Scenario C, when throttled load balancing algorithm is used, the results obtained are shown in Table3:

Table 3. Results of Throttled Load Balancing Algorithm

Parameter	Average (ms)	Min. (ms)	Max. (ms)
Total Response Time	57.68	34.5	87.01
Data Center Processing Time	7.24	0.02	25.05

The results obtained for the cost parameter after simulation of the senerio C is 1444.53$.

Comparing the scenario A, scenario B and scenario C having different algorithms, the overall response time and the data center processing time graphs obtained are shown in Fig6 and Fig7.

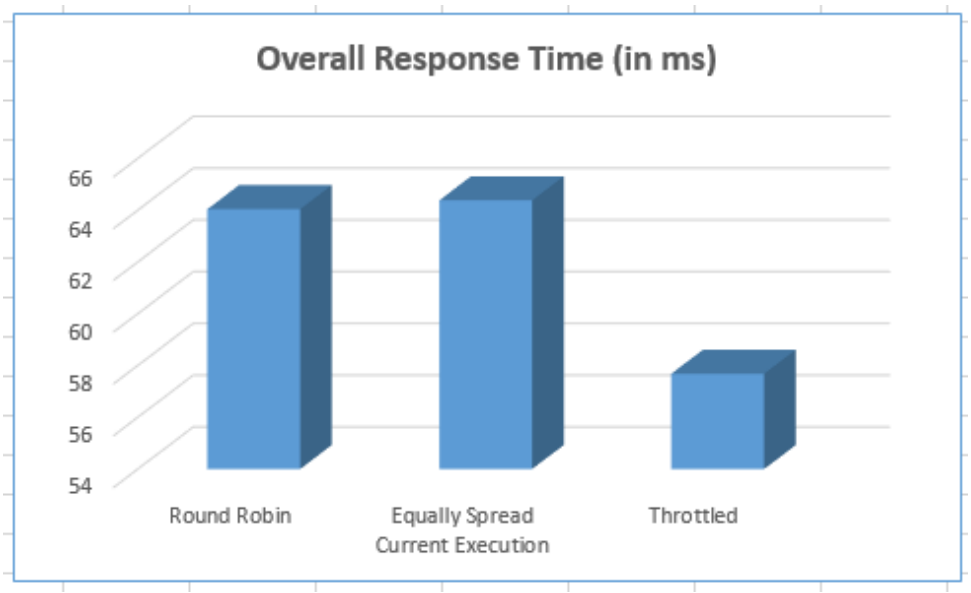

Figure 6. Overall Response Time of Load Balancing Algorithms

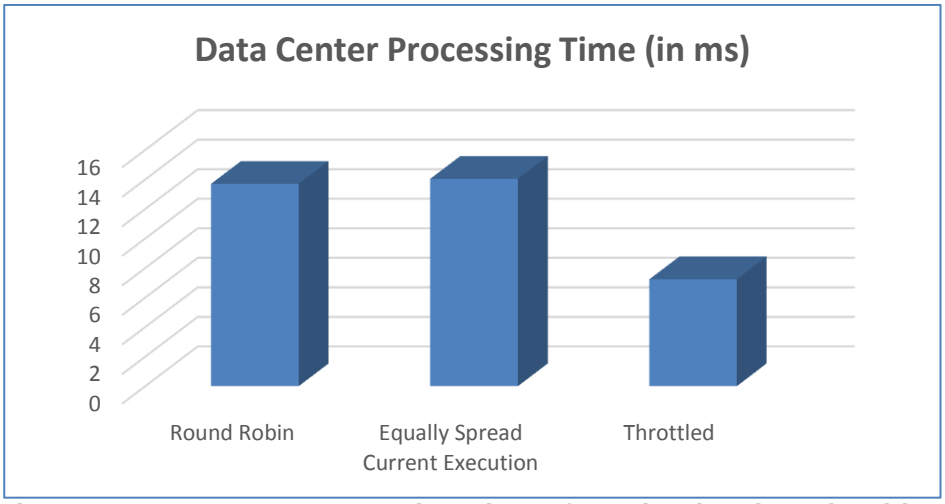

Figure 7. Data Center Processing Time of Load Balancing Algorithms

6. Conclusion & Future Scope

Comparing the results obtained from the different load balancing algorithms, the response time of throttled is good for six data center and six userbases as matched to round robin and equally spread current execution load balancing algorithms. Also, the processing time taken by the data center is good from both these algorithms mentioned above. Cost remains the same in all these algorithms. Seeing to the results, we can say throttled load balancing algorithm proves to be better than the other two algorithms.

The comparison is performed among the three algorithms that are taken into considerations. This work can be extended further using the different particle swarm optimization algorithms. Also, we can extend this too by using different service broker policies and then analyzing the results of the various load balancing techniques.

References

[1] M. Randles, D. Lamb, and A. Taleb-Bendiab. "A comparative study into distributed load balancing algorithms for cloud computing," in Proc. the 24th IEEE International Conference on Advanced Information Networking and Applications Workshops, 551-556, Fukuoka, Japan, (2011).
[2] Jeong H-Y, Park JH. An efficient cloud storage model for cloud computing environment. GPC 2012, (2012), 370–376.
[3] Kwok Y-K, Ahmad I. Static scheduling algorithms for allocating directed tasks graphs to multiprocessors. ACM Computing Surveys, 31(4), (1999), 406–471.
[4] Bhathiya Wickremasinghe, CloudAnalyst: A CloudSim-based Tool for Modelling and Analysis of Large Scale Cloud Computing Environments, a MEDC Project Report, (2009).
[5] Er. Simar Preet Singh and Er. Anshu Joshi. Cloud Computing, International Journal of Application or Innovation in Engineering & Management (IJAIEM), ISSN 2319 – 4847, Volume 3, Issue 3, (2014), March 2014.
[6] Lin C, Lu S. Scheduling scientific workflows elastically for cloud computing. IEEE CLOUD 2011, (2011), 746–747.
[7] Calheiros RN, Ranjan R, Beloglazov A, Rose CAFD. Buyya R. CloudSim: a toolkit for modeling and simulation of Cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and Experience, 41(1), (2011), 23–50.
[8] Chang R-S, Chang J-S, Lin S-Y. Job scheduling and data replication on data grids. Future Generation Computer Systems, 23(7), (2007), 846-860.
[8] Salehi MA, Buyya R. Adapting market-oriented scheduling policies for cloud computing. ICA3PP (1), (2010), 351–362.
[9] Zheng Q, Tham C-K, Veeravalli B. Dynamic load balancing and pricing in grid computing with communication delay, Journal of Grid Computing, 6(3), (2008), 239–253.

Authors

Simar Preet Singh, presently a research scholar at Thapar University, Patiala is also a Microsoft Professional. Apart from this, he is also having certifications like Microsoft Certified System Engineer (MCSE), Microsoft Certified Technology Specialist (MCTS) and Core Java. He had also undergone training programme for VB.Net and Cisco Certified Network Associates (CCNA). He has also worked with Infosys Limited for nearly two years and as an Assistant Professor at DAV University, Jalandhar for three years. He has presented many research papers in various National and International Conferences in India and abroad. His areas of interests include Cloud Computing, Bigdata and Network Security.

Anju Sharma completed her BIS in Computer Science (2003) and MScIT from Punjab University (2005) and Ph.D. in Grid Computing from Thapar University, Patiala (2009) and has over seven years of teaching and ten years of research experience. She is working as Associate Lecturer in Computer Science and Engineering Department, Thapar University, Patiala. Her research interests include Grid computing, Cloud computing and resource management challenges in Grids and Clouds. She has varied numbers of publications in International Journals and Conferences of repute. She is Senior Member of IACSIT (Senior Member of International Association of Computer Science and Information Technology) and professional member of ACM India, IEEE. She is an active member (TCM and Reviewer) of varied conferences.

Rajesh Kumar is currently a Professor in Thapar University, Patiala. He Obtained his M.Sc., M. Phil. and Ph. D. degree from IIT Roorkee. He has more than 20 years of UG & PG teaching and research experience. He has more than 80 research papers to his credit in various international and national journals and conferences. He is member of various professional bodies/societies of repute. His area of research and interest includes Cloud Computing, Wireless Networks and Software Engineering.