ISSN (online): 2349-784X

Analytical Study for Throttled and Proposed Throttled Algorithm of Load Balancing in Cloud Computing using Cloud Analyst

Slesha Nayak
Student
Department of Computer Science & Engineering
Narnarayan Shastri Institute of Technology, Jetalpur, India

Mr. Pragnesh Patel
Assistant Professor
Department of Computer Science & Engineering
Narnarayan Shastri Institute of Technology, Jetalpur, India

Abstract

Cloud Computing is an emerging technology and has attracted a lot of attention in both commercial and academic spheres. It is growing very fast and provides an alternative to conventional computing where all resources available via internet. It is new paradigm of large scale distributed computing which promise to offer subscription-oriented, enterprise-quality computing services to its users On-Demand bases in pay-per-use manner. One of important issue associated with this field is load balancing and task scheduling. Load balancing algorithms were investigated heavily in various environments; however, with Cloud environments, some additional challenges are present and must be addressed. Issues such as handling large scaled resources of Cloud require several techniques to optimize and streamline its operations. In this paper, comparative study is performed for existing throttle algorithm and proposed throttled algorithm of load balancing in cloud computing. Proposed throttled algorithm is implemented and tested. Both of these algorithms are compared in terms of Response time, Datacenter Request Servicing time and Cost in Cloud Analyst and Results prove the performance of proposed algorithm.

Keywords: Cloud computing, Load balancing algorithms, Virtual machine, Cloud Analyst, CPU Utilization, Loading Condition, Response time, Datacenter Request Servicing Time and Cost

I. INTRODUCTION

Cloud Computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g. network, servers, storage, application, and services) that can be rapidly provisioned and released with minimal management effort or services provider interaction, Defined by NIST [12]. The fundamental principle of Cloud Computing is to shift the computing from traditional desktop to the internet that is moving computation, services and data off-site to an external, internal, location transparent centralized contractor. This model is often referred as "pay-per-use model" that is users essentially rent virtual resources and pay for what they use [3]. According to the services provided by cloud, delivery models are of three types: Infrastructure as a Service (IaaS), The services are offered to users in the form of hardware requirements of its user to deploy their Virtual Machines (VMs). Platform as a Service (PaaS) which is a software platform for hosting application is already installed in an infrastructure and user uses this platform to develop their specific application, and Software as a Service (SaaS) is last level in which actual application is offered to users [2]. A Cloud Deployment Model is of four types. Public cloud is situated on the premises of the cloud provider, Private cloud is dedicated to particular organization, Community clouds offers services to organizations that have common functions and purposes and Hybrid cloud is a Combination of public, private and community clouds forms [2].

There are several challenges in Cloud Computing that need to be resolved before exploiting the features this technology. Some challenges include security issues [2], legal and compliant issues [14], load balancing [2], reliability [2], owner ship [2], performance and QOS [14], interoperability issues [14], data management issues [14], multi-platform support [2]. Load balancing is a methodology to distribute workload across multiple computers, or other resources over the network links to achieve optimal resource utilization, maximize throughput, minimum response time, and avoid overload [3]. Load Balancing is one of the primary concerns in Cloud Computing. Few challenges must be taken in to account while implementing load balancing for optimal solution are spatial distribution of the cloud node, Storage/Replication, Algorithm Complexity and Point of Failure [1].

As we know cloud platform can be quickly scaled up and down at any point of time. So the numbers of user's requests can join to and leave from the cloud during the execution of the applications. In this dynamic environment an efficient load balancing is required to minimize the response time, lower network congestion, avoid the interruption of services, limited energy consumption and provides high availability which means continuity of services when components becomes non-responsive [14].
In this paper, Analytical study is performed for Proposed Throttled algorithm, and it is compared with existing throttle algorithm. Proposed algorithm take decision for VM allocation to any request based on available VMs, its CPU utilization and expected response time for jobs in execution [11]. Then assign all incoming requests uniformly among the available VMs in an efficient way.

The rest of this paper organized as follows. Existing throttled load balancing algorithm and its limitations. Propose changes in existing throttled algorithm and its steps, Proposed System Model, Experimental tool, Parameter Setup and Results. Existing and Proposed algorithms are compared in terms of Response time, Datacenter Request Servicing time and cost in Cloud Analyst and Results prove the performance of proposed algorithm.

II. RELATED WORK

A. Existing Throttled Load Balancing Algorithm:

Throttled load balancing algorithm is relied on status of VM. Here status is referred as allocation of VM, allocated or not. This information is stored in an index table at load balancer. This index table containing two parameters, one is to identify an individual VM, so it's ID and status of that VM in the form of (Available or Busy). At start, all VM set to available mode **Error! Reference source not found.**. First of all client request received by data centre, then it will forward them to load balancer to find an appropriate VM for user request. Now which VM can take this load of new job will be decided by load balancer. Load balancer will start scanning the index table **Error! Reference source not found.** from the top, and search for an available VM. If any of the available VM is found from the index table, than id of that VM will be notified to the datacenter controller for request assignment. If datacenter successfully allocate that VM id to specified request than it will notify the success of its operation to load balancer and accordingly, load balancer updates its index table. If datacenter controller found any trouble during the allocation it will give negative feedback notification within specific time duration, which leads to no updates in index table. If a situation occurs when all VM having status busy than datacenter controller receives -1 from the load balancer **Error! Reference source not found.**. In this situation datacenter will start queuing user requests at its own pool. When job allocated to VM completed the VM itself notify job done to datacenter controller, and it will notify the same to load balancer, and Load balancer will make respective changes in index table. The total execution time is estimated in three phases **Error! Reference source not found. Error! Reference source not found.**. In the first phase the formation of the VMs and they will be idle waiting for the scheduler to schedule the jobs in the queue, once jobs are allocated, the VMs in the cloud will start processing, which is the second phase, and finally in the third phase the cleanup or the destruction of the VMs. The throughput of the computing model can be estimated as the total number of jobs executed within a time span without considering the VM formation time and destruction time.

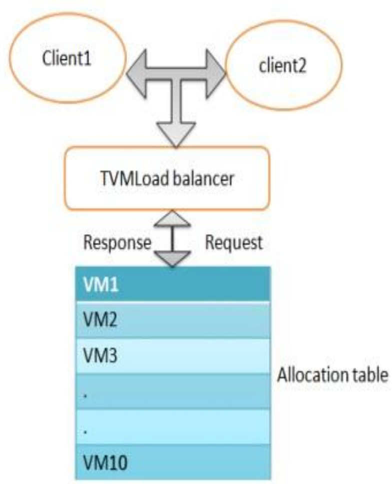

Fig. 1: VM Placement in Throttle algorithm

B. Limitations of Existing Methodology:

Throttle load balancing algorithm maintains a hash table for current state of VM (Busy/Available), which help a lot in VM placement. Among all described algorithm above throttle load balancing algorithm have best approach for load balancing. But it works properly only when the hardware configuration of all VMs of data center had similar hardware Configuration.

C. Proposed Model with Improved Throttled Algorithm:

The proposed load balancing model will use improved throttle algorithm. This improved throttle algorithm works well even though underlying capacity of each VM is different because the hardware configuration of VMs is different. So improved throttle algorithm is taking decision of VM selection with hash table with more parameters such as expected response time and loading condition. Now Expected response time can be calculated using CPU utilization of VM. Using improved throttled load balancing algorithm with less overhead, results better VM allocation and increased number of user request handling, thus reducing the rejection in the number of requests arrived at datacenter of cloud.

III. PROPOSED SYSTEM MODEL

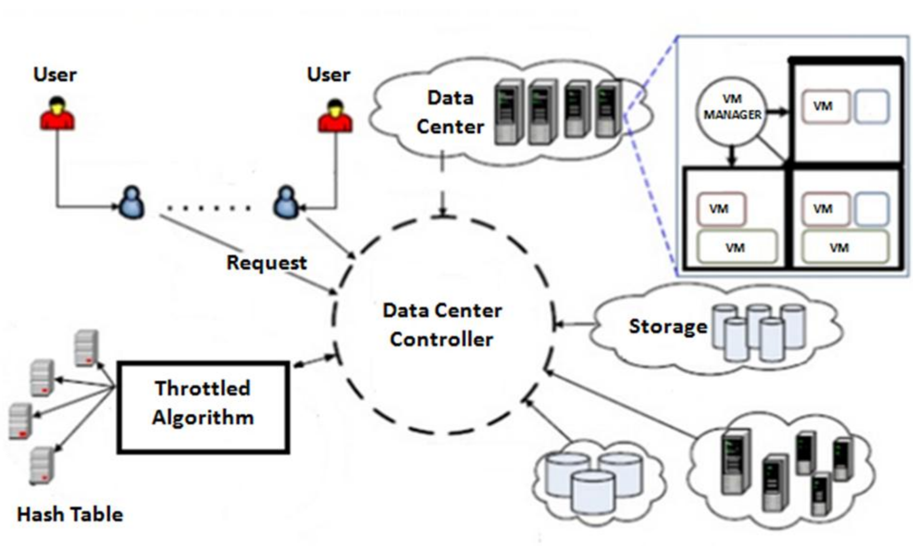

Fig. 2: Proposed System Model

A. *Proposed Algorithm*

Input:
- Data centre requests r1,r2,….., rn
- Available VMs vm1,vm2, ,vmn

Output:
- Data centre requests r1,r2,……,rn are allocated available VMs vm1,vm2,………,vmn

1) Steps:
1) The improved throttled algorithm maintains a hash map table of all the available VMs which their current state and the expected response time. This state may be available or busy. At the beginning, all the VMs are available.
2) When data centre controller receives a request then it forwards that request to the improved throttled load balancer. The improved throttled load balancer is responsible for the VM allocation. So that the job can be accomplished.
3) The improved throttled algorithm scans the hash map table. It checks the status of the available VMs.

2) If A VM With Least Load And The Minimum Response Time Is Found.
- Then the improved throttled algorithm sends the VM id of that machine to the data centre controller
- Data centre controller sends a request to that VM
- Data centre controller sends a notification of this new allocation to the improved throttled
- The improved throttled algorithm updates the hash map index accordingly

3) If a VM is not Found Then the Improved Throttled Algorithm Returns -1 To the Data Centre Controller

4) When the VM finishes the request.
- The data centre controller sends a notification to improve throttled that the VM id has finished the request.
- improved throttled modifies the hash map table accordingly

5) If there are more requests then the data centre controller repeats step 3 for other VMs until the size of the hash map table is reached. Also of the size of hash map table is reached then the parsing starts with the first hash map index.

The proposed algorithm will take decision on bases of few run-time parameters of the system such as expected Response Time of VM and loading condition. This expected response time of each VM is purely based on no of jobs allocated to it with respect to its processing capabilities. In heterogeneous cloud environment it can be calculated using following formula **Error! Reference source not found.**:

$$\text{Response Time} = Fint - Arrt + TDelay \qquad (1)$$

Here Arrt is the arrival time of user request and Fint is the finish time of user request and the transmission delay can be determined using the following formula **Error! Reference source not found.**:

$$TDelay = Tlatency + Ttransfer \qquad (2)$$

Where TDelay is the transmission delay, T latency is the network latency and T transfer is the time taken to transfer the size of single request from source location to destination **Error! Reference source not found.**.

$$Ttransfer = D / Bwperuser \quad (3)$$
$$Bwperuser = Bwtotal / Nr \quad (4)$$

Where Bwtotal is the total available bandwidth and Nr 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 Nr **Error! Reference source not found.**.

IV. EXPERIMENTAL TOOL

The Cloud Analyst is GUI based simulation tool built on top of CloudSim tool kit, by extending CloudSim functionality with the introduction of concepts that model Internet and Internet Application behaviors[11].

Development of large-scale applications in the cloud using the simulator is quite economical and easy. Cloud Analyst is basically made for evaluating performance and cost of large-scale geographically distributed cloud system that is having huge user workload based on different parameters. It has very attractive GUI and huge flexibility to configure any geographical distribute system such as setting hardware parameters (storage, main memory, bandwidth limit, network delays etc.) of a virtual machine or data center [22]. New service brokering policy can be added easily that control the users of any geographical location based on services done by which Data Center at any particular given time **Error! Reference source not found.**. In Cloud Analyst, analysis can be done repeatedly and can generate output in the form of chart and table that summarize the huge amount user, system statistic during the simulation time **Error! Reference source not found.**.

Cloud Analyst simulator allows to create various experimentation scenarios on cloud having different parameters. Being a modeler just focus on applying different simulation strategies, it helps in generating the knowledge base for different situations. Performing series of simulation operations with slight change in any of the parameters is quite easy and quick in Cloud Analyst.

B. Components of Cloud Analyst:

- GUI Package **Error! Reference source not found.**: It is responsible for proving the graphical user interface.
- Simulation **Error! Reference source not found.**: This component is for creating and executing the simulation.
- UserBase **Error! Reference source not found.**: This component models a user base and generates traffic representing the users.
- DataCenterController **Error! Reference source not found.**: This component controls the data center activities.
- Internet: This component models the Internet and implements the traffic routing behavior.
- VmLoadBalancer **Error! Reference source not found.**: This component models the load balance policy used by data centers when serving allocation requests.
- CloudAppServiceBroker **Error! Reference source not found.**: This component models the service brokers that handle traffic routing between user bases and data centers.

C. Cloud Analyst Simulation Interface and Experimental Parameter Set up

In order to analyze specific load balancing algorithm, configuration of various components and its parameters required to set in Cloud Analyst simulation tool. The Home Screen of Cloud Analyst Simulator is shown in Fig 3.

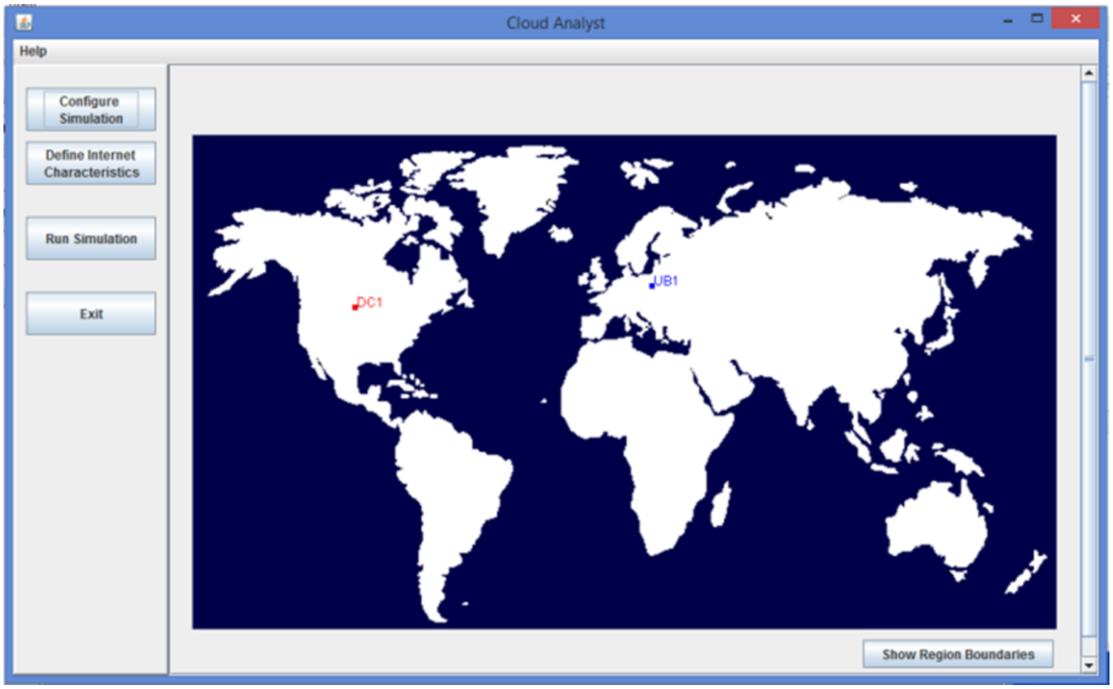

Fig. 3: Cloud Analyst Home Screen

Configuration Simulation divided in three parts, Main Configuration where we can setup parameter such as simulation duration, user base, service broker policy, application configuration. Fig 4 shows the parameters values we have choose for our experiment.

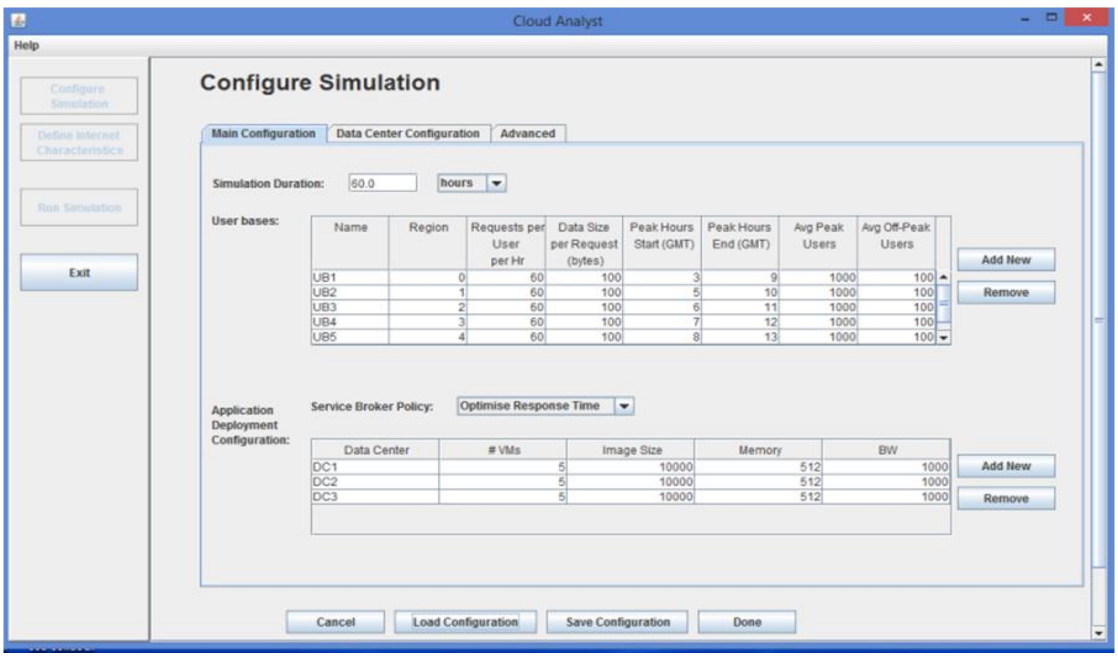

Fig. 4: Main Configuration Cloud Analyst

The brokering policies we take in to the consideration to compare existing and proposed throttled algorithm is Optimize Response Time. Simulation duration is set to 60 minutes.

Various scenarios are created by assigning different component values for User base and Deployment configuration.

In first Scenario there are three Datacenter and Ten User Base is created as shown in Fig 4. And in second two Datacenter and five user base is formed for analysis.

Data center configuration has to be done before main configuration as datacenter created in this tab will be displayed for selection in main configuration. Datacenter region, architecture it's working on, operating system, cost for accounting, and processing elements such as cores, memory, processor speed and vm-policy can be configured as shown in Fig 5.

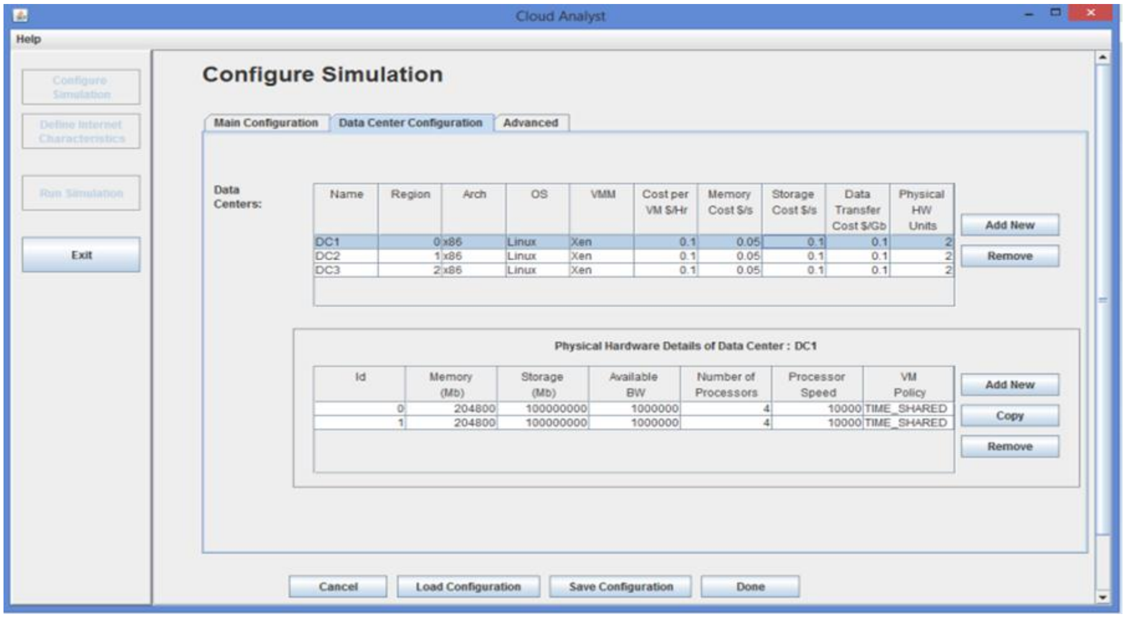

Fig. 5: Data center Configuration

In our experimental setup each datacenter's parameter values such as processing architecture is set to X86, operating system is Linux, Virtual Machine Manager is set to Xen, cost and no of physical hardware unit.

For each datacenter Physical Hardware properties section where we have set the VM memory 204800 MB, Storage 1TB, No of Processor 4 and VM policy TIME_SHARED.

Advanced simulation parameters such as user grouping factor (UGF), request grouping factor (RGF), Executable Instruction Length (EIL) and load balancing policy for VM can be selected for this tab as shown in Fig 6. Loading and saving configurations also possible for simulation.

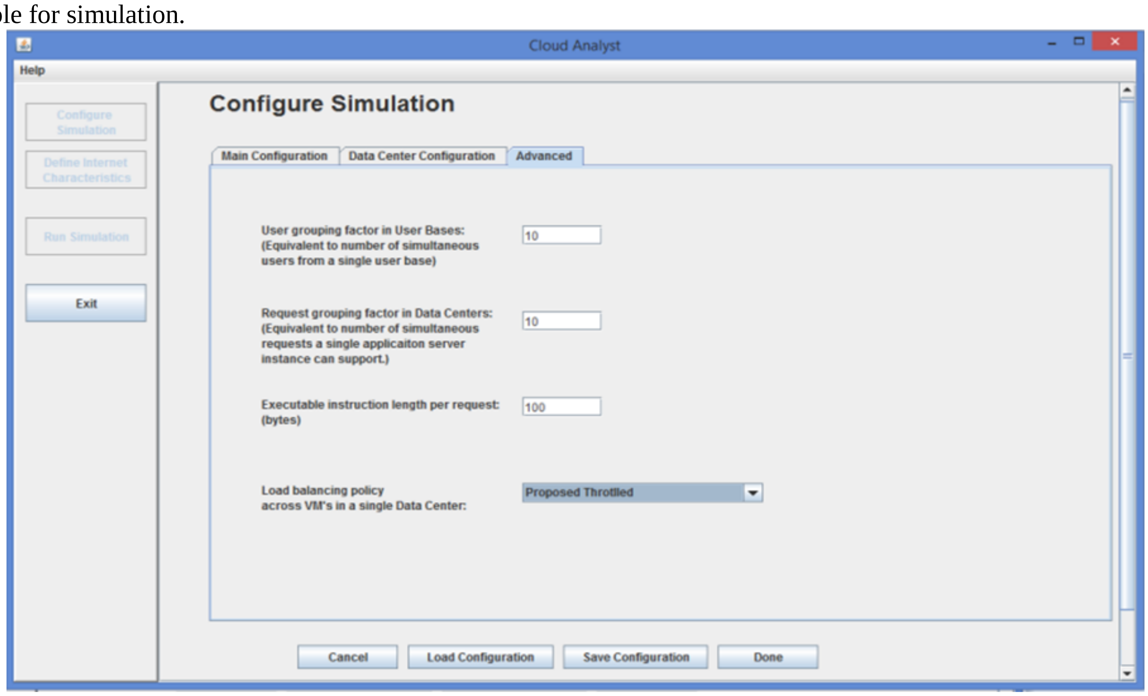

Fig. 6: Advance Configurations

For our experiment we can select different load balancing policy from the drop-down. As we have created new policy for Improved Throttle Algorithm we can select it from propped system simulation.

For both scenarios we have set UGF to 10, RGF to 10 and EIL to 100 bytes.

V. RESULTS

A. Comparison of Existing Throttle Algorithm and Proposed Throttle Algorithm

Table 1:
Comparison Table of Existing Throttle and Proposed Throttle Algorithm

Scenario 1								
Service Broker Policy : Optimize Response Time								
No. of Users	*10*				*UGF*	*RGF*	*Executable instruction length per Req.*	
No. of Data Centers :	*3*				*10*	*10*	*100*	
Response Time								
Existing Throttle Algorithm					*Proposed Throttle Algorithm*			
User Base	*Avg*	*Min*	*Max*		*User Base*	*Avg*	*Min*	*Max*
UB1	*49.842*	*40.121*	*61.127*		*UB1*	*49.84*	*39.837*	*60.611*
UB2	*50.269*	*36.62*	*62.623*		*UB2*	*50.205*	*36.62*	*62.62*
UB3	*50.172*	*38.356*	*61.364*		*UB3*	*50.144*	*38.36*	*61.364*
UB4	*50.208*	*39.871*	*62.872*		*UB4*	*50.2*	*39.871*	*60.87*
UB5	*50.011*	*38.613*	*61.355*		*UB5*	*50.005*	*38.6*	*61.35*
UB6	*50.3*	*36.621*	*61.872*		*UB6*	*50.3*	*36.615*	*61.87*
UB7	*50.121*	*39.877*	*61.122*		*UB7*	*50.051*	*38.895*	*61.12*
UB8	*49.901*	*39.12*	*61.124*		*UB8*	*49.889*	*39.1*	*61.12*
UB9	*50.126*	*40.126*	*60.623*		*UB9*	*50.121*	*40.126*	*60.619*
UB10	*50.004*	*40.121*	*61.127*		*UB10*	*50.067*	*40.624*	*61.127*
Average :	*50.0954*	*38.9446*	*61.5209*		*Average :*	*50.0822*	*38.8648*	*61.2671*
Data Center Request Servicing Time								
Existing					*Proposed*			
Data Center	*Avg*	*Min*	*Max*		*Data Center*	*Avg*	*Min*	*Max*
DC1	*0.472*	*0.01*	*0.875*		*DC1*	*0.469*	*0.01*	*0.871*
DC2	*0.485*	*0.01*	*0.885*		*DC2*	*0.48*	*0.005*	*0.88*
DC3	*0.49*	*0.009*	*0.886*		*DC3*	*0.485*	*0.007*	*0.885*
Cost								
Existing					*Proposed*			
Data Center	*VM Cost*	*Data Transfer Cost*	*Total*		*Data Center*	*VM Cost*	*Data Transfer Cost*	*Total*
DC1	*0.5*	*0.265*	*0.765*		*DC1*	*0.5*	*0.255*	*0.755*
DC2	*0.5*	*0.198*	*0.698*		*DC2*	*0.5*	*0.192*	*0.692*
DC3	*0.5*	*0.199*	*0.699*		*DC3*	*0.5*	*0.195*	*0.695*
Grand Total :	*1.5*	*0.662*	*2.162*		*Grand Total :*	*1.5*	*0.642*	*2.142*

B. *Min and Max Response time Summary of Existing and Proposed Throttle Algorithm where Service broker Policy is: Optimized Response Time, No of Data Center 3 and User Base 10*

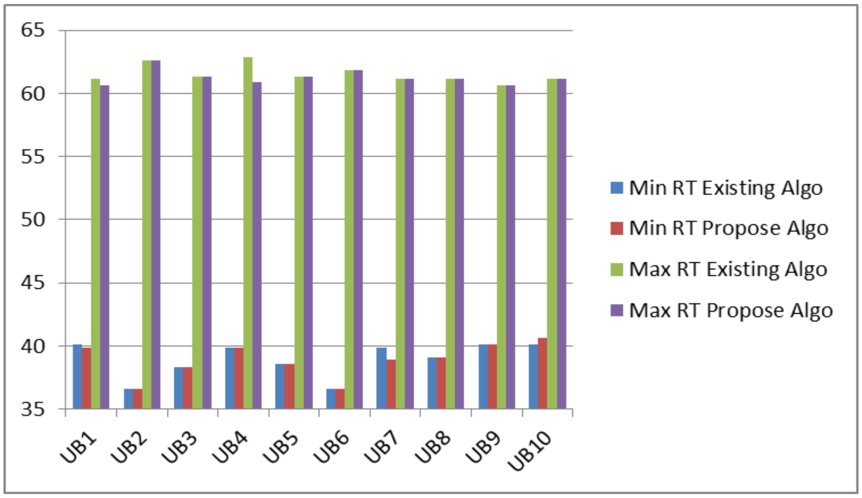

Fig. 7: Min and Max Response time Summary of Existing and Proposed Throttle Algorithm where SBP is ORT

C. *Avg Response time Summary of Existing and Proposed Throttle Algorithm where Service broker Policy is: Optimized Response Time No of Data Center 3 and User Base 10*

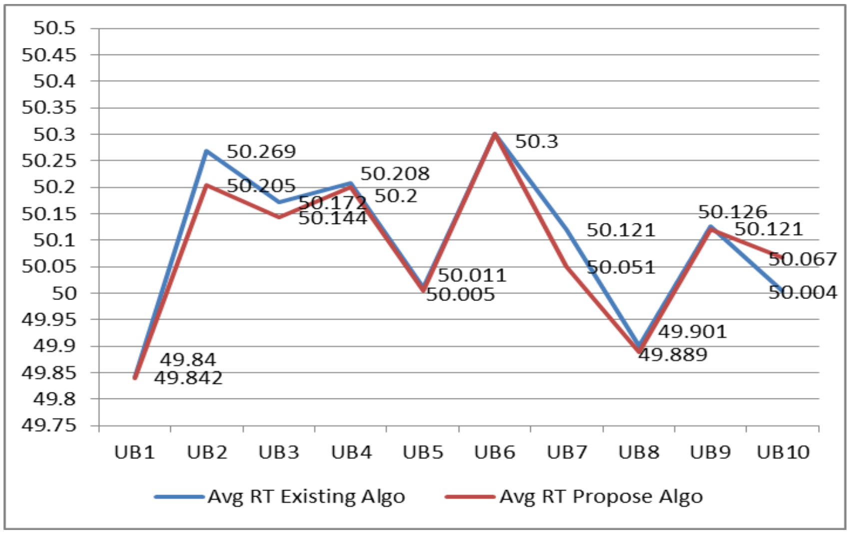

Fig. 8: Avg Response time Summary of Existing and Proposed Throttle Algorithm where SBP is ORT

Table 2:
Comparison Table of existing Throttle and Proposed Throttle Algorithm

Scenario 2								
Service Broker Policy : Optimize Response Time								
No. of Users		*5*			*UGF*	*RGF*	*Executable instruction length per Req.*	
No. of Data Centers :		*2*			*10*	*10*	*100*	
Response Time								
Existing					*Proposed*			
User Base	*Avg*	*Min*	*Max*		*User Base*	*Avg*	*Min*	*Max*
UB1	*49.89*	*39.875*	*59.853*		*UB1*	*49.892*	*39.87*	*59.847*
UB2	*50.269*	*40.362*	*60.358*		*UB2*	*50.17*	*40.362*	*60.356*
UB3	*50.27*	*37.872*	*61.381*		*UB3*	*50.267*	*36.872*	*61.373*

UB4	50.153	40.423	58.871		UB4	50.054	40.356	58.611
UB5	50.106	40.152	60.642		UB5	50.119	40.122	60.621
Average :	50.1376	39.7368	60.221		Average :	50.1004	39.5164	60.1616
Data Center Request Servicing Time								
Existing					Proposed			
Data Center	Avg	Min	Max		Data Center	Avg	Min	Max
DC1	0.471	0.01	0.875		DC1	0.469	0.01	0.871
DC2	0.486	0.01	0.885		DC2	0.479	0.005	0.88
Cost								
Existing					Proposed			
Data Center	VM Cost	Data Transfer Cost	Total		Data Center	VM Cost	Data Transfer Cost	Total
DC1	0.504	0.201	0.705		DC1	0.502	0.192	0.694
DC2	0.504	0.142	0.646		DC2	0.502	0.128	0.63
Grand Total :	1.008	0.343	1.351		Grand Total :	1.004	0.32	1.324

D. Min and Max Response time Summary of Existing and Proposed Throttle Algorithm where Service broker Policy is: Optimized Response Time, No of Data Center 2 and User Base 5

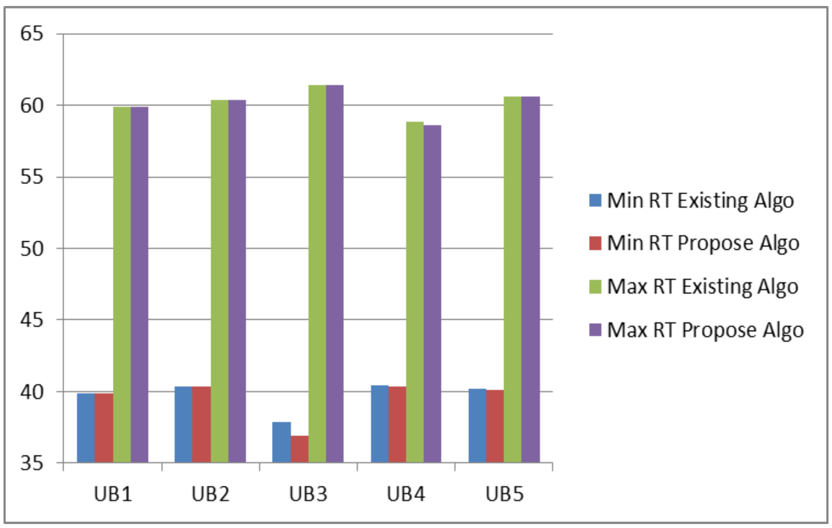

Fig. 9: Min and Max Response time Summary of Existing and Proposed Throttle Algorithm where SBP is ORT

E. Avg Response time Summary of Existing and Proposed Throttle Algorithm where Service broker Policy is: Optimized Response Time, No of Data Center 2 and User Base 5

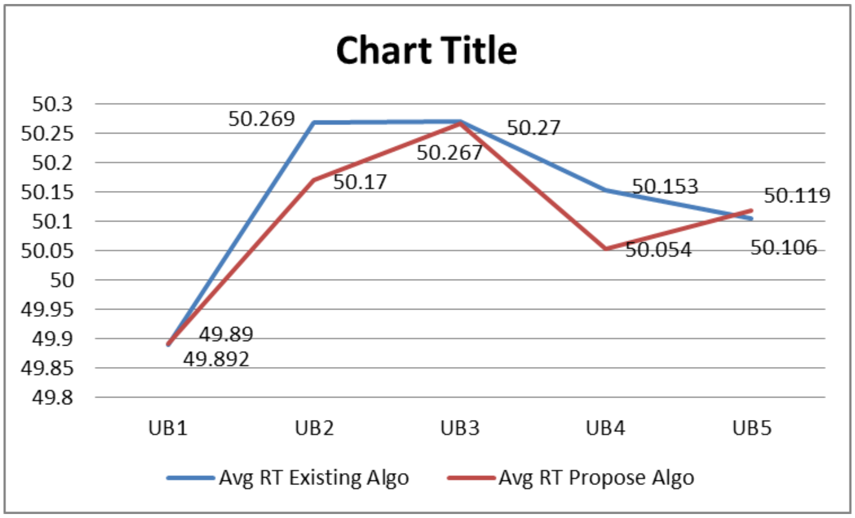

Fig. 10: Avg Response time Summary of Existing and Proposed Throttle Algorithm where SBP is ORT

VI. CONCLUSIONS AND FUTURE ENHANCEMENT

Comparative and analytical study of existing and proposed throttle algorithm is performed. It proves that it is possible and beneficiary to include more load balancing parameters to check the availability of VM for precise decision. Implementation of proposed throttle algorithm reduces response time, datacenter request servicing time as well as cost.

Future work is to implement the proposed algorithm on real cloud environment such as open-nebula where VMs having different hardware configuration.

REFERENCES

[1] Klaithem Al Nuaimi, Nader Mohamed, Mariam Al Nuaimi, Jameela Al-Jaroodi. " A Survey of Load Balancing in Cloud Computing: Challenges and Algorithms." IEEE Second Symposium on Network Cloud Computing and Applications (2012) 137-142.
[2] Suriya Begum, Dr. Prashanth C.S.R. "Review of Load Balancing in Cloud Computing." International Journal of Computer Science Issues-IJCSI (January 2013) 343-352.
[3] Soumya Ray, Ajanta De Sarkar. "EXECUTION ANALYSIS OF LOAD BALANCING ALGORITHMS IN CLOUD COMPUTING ENVIRONMENT. " International Journal on Cloud Computing: Services and Architecture-IJCCSA (October 2012) 1-13.
[4] Shreyas Mulay, Sanjay Jain. " Enhanced equally distributed load balaning algorithm for cloud computing." International Journal of Reasearch in Engineering and Technology-IJRET (Jun 2013) 976-980.
[5] Borja Sotomayor, Rubén S. Montero, Ignacio M. Lorente, Ian Foster. " Virtual Infrastructure Management in Private Cloud." IEEE Internet Computing (2011) 14-22.
[6] B.Radojevic, M.Zagar. "Analysis of issues with load balancing algorithms in hosted (cloud) environments." MIPRO, 2011 Proceedings of the 34th International Convention (2011) 416-420.
[7] Nitika, Shaveta, Gaurav Raj. "Comparative Analysis of Load Balancing Algorithms in Cloud Computing. " Iternational Journal of Advance Reasearch in Computer Engineering & Technology-IJARCET (May 2012) 120-124.
[8] Mala Kalra, Gulshan Soni. "Novel Aprroach for Load Balancing in cloud data center." IEEE international Advance Computing Conference-IACC (2014) 807-812.
[9] Xiaona Ren, Rongheng Lin, Hua Zou. "A dynamic load balancing strategy for cloud computing platform based on exponential smoothing forecast." IEEE International Conference on Cloud Computing and Intelligent Systems-CCIS (Sep 2011) 220-224.
[10] Tejinder Sharma, Vinay Kumar Banga. " Efficient and Enhanced Algorithm in Cloud Computing." International Journal of Soft Comuting and Engineering-IJSCE (Mar 2013) 385-390.
[11] Bhathiya Wickremasinghe. "CloudAnalyst: A CloudSim-based Tool for Modelling and Analysis of Large Scale Cloud Computing Environments." MEDC Project Report (Sep 2009) 1-44.
[12] Peter Mell, TimothyGrance. " The NSIT definition of Cloud." NIST Special Publication (2011) 145.
[13] N. S. Raghava, Deepti Singh. "Comparative Study on Load Balancing Techniques in Cloud Computing." OPEN JOURNAL OF MOBILE COMPUTING AND CLOUD COMPUTING (Aug 2014) 18-25.
[14] Raza Abbas Haidri, C. P. Katti, P. C. Saxena. "A Load Balancing Strategy for Cloud Computing Environment." International Conference on Signal Propagation and Computer Technology (ICSPCT)IEEE (July 2014) 636-641.
[15] Rajkumar Buyya, Rodrigo N. Calheiros, Rajiv Ranjan, Anton Beloglazov, Cesar A. F. De Rose. "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms." Software Practice and Experience (Aug 2010) 23-50.
[16] Shoja, H., Nahid, H. and Azizi, R. "A comparative survey on load balancing algorithms in cloud computing. " International Conference on Computing Communication and Networking Technologies - ICCCNT (July 2014) 1-5.
[17] Vasudha Arora, S S Tyagi. "Performance Evaluation of Load Balancing Policies across Virtual Machines in a Data Center." International Conference on Reliability, Optimization and Information Technology - ICROIT (Feb 2014) 384-387.
[18] Bhushan Ghutke, Urmila Shrawankar. "Pros and Cons of Load Balancing Algorithm for Cloud Computing." Information Systems and Computer Networks -ISCON (March 2014) 123-127.

[19] Veerawali Behal, Anil Kumar. "Cloud computing: Performance analysis of load balancing algorithms in cloud heterogeneous environment. " Confluence The Next Generation Information Technology Summit (Confluence),IEEE (Sep 2014) 200-205.
[20] Viswas Bhagwaiya, Sandeep K Raghuvanshi. "Hybrid approach using throttled and ESCE load balancing algorithms in cloud computing." Green Computing Communication and Electrical Engineering -ICGCCE (Mar 2014) 1-6.
[21] Subhadra Bose Shaw, Dr. A.K. Shingh. "A survey on scheduling and load balancing techniques in cloud computing environment." International Conference on Computer and Communication Technology-ICCCT (Sep 2014) 87-95.
[22] Arif Ahmed, Abadhan Sumya Sabyasachi. "Cloud computing simulators: A detailed survey and future direction." International Advance Computing Conference-IACC (Feb 2014) 866-872.
[23] Cloud computing From Wikipedia, the free encyclopedia. [Online] http://en.wikipedia.org/wiki/Cloud_computing..
[24] CloudSim. [Online] http://www.cloudbus.org/CloudSim.
[25] Chhagan Athode, Bhagyashri Kale, Neha Alone, Khusboo Yadav, Ankush Penshanwar, Nikhil Gondane. "DATA INTEGRITY FOR ANALYSIS OF THE CLOUD COMPUTING SECURITY. " International Journal of Computer, Information Technology & Bioinformatics-IJCITB (Apr 2013) 45-49.
[26] Mayanka Katyal, Atul Mishra. "A Comparative Study of Load balancing Algorithms in Cloud Computing Environment" International Journal of Computer Science and Communication Engineering-IJCSCE (Feb 2014) 1-8.
[27] Asitha Micheal, Jalpa Mehta. "Load Balancing Techniques in Cloud Computing. " International Journal on Recent and Innovation Trends in Computing and Communication (Oct 2014) 3038-3044.
[28] SHANTI SWAROOP MOHARANA, RAJADEEP D RAMESH, DIGAMBER POWAR. " ANALYSIS OF LOAD BALANCERS IN CLOUD COMPUTING." International Journal Computer Science Engineering-IJCSE (May 2013) 101-108.
[29] Reddy, Shridhar G.Domanal and G.Ram Mohana. "Load Balancing in Cloud Computing Using Modified Throttled Algorithm . " IEEE International Conference on Cloud Computing in Emerging Markets-CCEM (Oct 2013) 1-5.
[30] B.KalaiSelvi, Dr.L.Mary Immaculate Sheela. "A Survey of Load Balancing Algorithms using VM. " International Journal of Advancements in Research & Technology-IJOART (Aug 2014) 68-76.
[31] Sukhvir Kaur, Supriya Kinger. " Review on Load Balancing Techniques in Cloud Computing Environment . " International Journal of Science and Research-IJSR (June 2014) 2499-2504.