pISSN 1976-913X
eISSN 2092-805X

J Inf Process Syst, Vol.9, No.3, September 2013 http://dx.doi.org/10.3745/JIPS.2013.9.3.379

Round Robin with Server Affinity: A VM Load Balancing Algorithm for Cloud Based Infrastructure

Komal Mahajan*, Ansuyia Makroo* and Deepak Dahiya*

Abstract—Cloud computing is an evolving computing paradigm that has influenced every other entity in the globalized industry, whether it is in the public sector or the private sector. Considering the growing importance of cloud, finding new ways to improve cloud services is an area of concern and research focus. The limitation of the available Virtual Machine Load balancing policies for cloud is that they do not save the state of the previous allocation of a virtual machine to a request from a Userbase and the algorithm requires execution each time a new request for Virtual Machine allocation is received from the Userbase. This problem can be resolved by developing an efficient virtual machine load balancing algorithm for the cloud and by doing a comparative analysis of the proposed algorithm with the existing algorithms.

Keywords—Virtual Machine (VM), Server affinity, VM load balancer, CloudAnalyst, Data center, Cloudlet

1. INTRODUCTION

The current computing era is that of cloud computing or cloud. In the remaining sections of the paper the terms *cloud computing* and *cloud* have been used interchangeably. Irrespective of the developments in the IT industry, the one evolving paradigm that has influenced every other entity in the globalized industry, whether it is in the public sector or the private sector, is cloud computing. "The Obama administration's budget for the 2013 Fiscal Year (FY) is clear that the deployment of cloud computing solutions will remain a priority for U.S. Government Departments and Agencies moving forward" [1]. Even government agencies are supporting the growth of the cloud computing. The latest trend shows a growing number of small and medium scale businesses are moving to cloud. The number of service providers are growing and the cost of services are decreasing. Considering the growing importance of cloud, finding new ways to improve cloud services is an area of concern and research focus.

Cloud computing can be defined as follows:

"*Cloud computing is the delivery of computing as a service rather than a product, whereby shared resources, software, and information are provided to computers and other devices as a utility (like the electricity grid) over a network (typically the Internet)* [2]. "

"*Cloud computing is defined as a type of parallel and distributed system consisting of a collec-*

Manuscript received October 23, 2012; first revision January 30, 2013; accepted July 23, 2013.
Corresponding Author: Komal Mahajan
* Dept. of CSE & ICT, Jaypee University Of Information Technology, Waknaghat, HP, India (komal.mahajan@juit.ac.in, ansuyia.makroo@juit.ac.in, deepak.dahiya@juit.ac.in)

tion of inter-connected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements established through negotiation between the service provider and consumers [3]".

Sun Microsystems (now acquired by Oracle) [4] "*takes an inclusive view that there are many different types of clouds like public cloud, private cloud, and hybrid cloud models. Many different applications can be built by using these clouds.*"

A Cloud consists of a number of clusters, which are further divided into a number of nodes, and each node consists of a number of VMs. The requests are actually deployed on the VMs. When a cloud receives a request from a user from a particular Userbase, the VM load balancing algorithms handle the task of allocating the VMs to the request. The currently available algorithms do not save the state of the previous allocation of the VMs to a request from a given Userbase. As such, every time a request is received from the same Userbase, the algorithm needs be run again, which increases the total response time and processing time of the requests. This research work aims to reduce the total response time and processing time of Userbase requests by proposing Round Robin with server affinity VM load balancing algorithm. The generalized overview of the proposed algorithm is depicted in Figure 1.

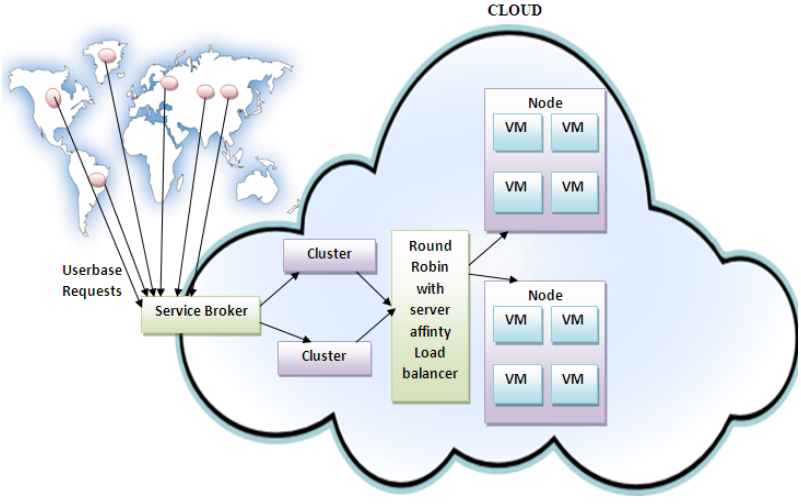


Fig. 1. Overview of the Round Robin with server affinity VM load balancing algorithm

1.1 Motivation and Problem Statement

With the ever increasing number of cloud service providers, rising competition, and the growing popularity of cloud, it is extremely important for cloud providers to improve their services by improving the operational cost (response time and the processing time of requests). Hence, to satisfy the service level objectives (SLOs) [13], it is very important to improve the low level requirements such as the CPU and memory requirements and the response time of the Userbase requests. One of the ways to improve this is by providing efficient VM load balancing policies. However, there are only a limited number of VM load balancing policies available at present. Also, in the existing VM load balancing strategies, an algorithm has to be run every time a new

request for VM allocation is received from the Userbase. This is because the existing VM load balancing strategies do not save the previous allocation state of a VM to a request from a given Userbase. However, in case of the proposed strategy, a hashmap is used to store the entries for the last VM allocated to a request from a given Userbase. Thus, when a request is received from a Userbase, if an entry for the given Userbase exists in the hashmap and if the particular VM is available, then there is no need to run the VM allocation algorithm, which in turn saves a significant amount of time.

The above discussion leads us to the following problem definition i.e. "To develop an efficient VM load balancing algorithm for the cloud and to do a comparative analysis of the proposed algorithm with the existing algorithms."

The problem definition leads us to the broad objectives that are summarized below:

- To study the existing VM load balancing algorithms.
- To propose an efficient algorithm for VM load balancing.
- To implement the algorithm on CloudAnalyst and to analyze its performance
- Comparison of the proposed algorithm with the existing algorithms on identified parameters

The rest of the paper is organized as follows: Section 1 discusses the introduction of the proposed work. Section 2 includes a related study on available cloud simulators and the VM Load Balancing algorithms. Section 3 gives a detailed description of the proposed algorithm (i.e., a Round Robin with server affinity. Section 4 describes the experimental setup and the simulation and parameter configuration. Section 5 discusses the simulation of the existing VM load balancer (i.e. the Round Robin Algorithm). Section 6 discusses the simulation of the proposed VM load balancer (i.e., a Round Robin with Server affinity). Section 7 includes the comparative analysis of the Round Robin and the Round Robin with server affinity VM load balancing algorithms. Section 8 includes the conclusion and future work that needs to be carried out.

2. RELATED STUDY

Studying and analyzing the proposed algorithm on a real cloud scenario is extremely difficult. Also, considering the budget constraints, the most feasible option is to study and analyze the proposed algorithm by using simulation. As such, we used CloudAnalyst to study and analyze the proposed algorithm. CloudAnalyst [5] is a cloud simulation tool that supports visual modeling and the simulation of large-scale applications that are deployed on Cloud Infrastructure. CloudAnalyst, which is built on CloudSim [6], allows for the description of application workloads, including information on the geographic location of users generating traffic and the location of data centers, the number of users and data centers, and the number of resources in each data center. Using this information, CloudAnalyst generates information about the response time of requests, the processing time of requests, virtual machine cost, and total data transfer cost. It is based on top of mature simulation frameworks such as SimJava [7,8] and CloudSim [6]. CloudSim framework [6] is in turn built on top of GridSim framework [9]. The GridSim toolkit is a Java based simulation toolkit that supports the modeling and simulation of heterogeneous Grids.

Figure 2 gives the overview of generalized architecture of the cloud. The Data center Controller [11] uses a VM Load Balancer to determine which VM should be assigned the next Cloudlet

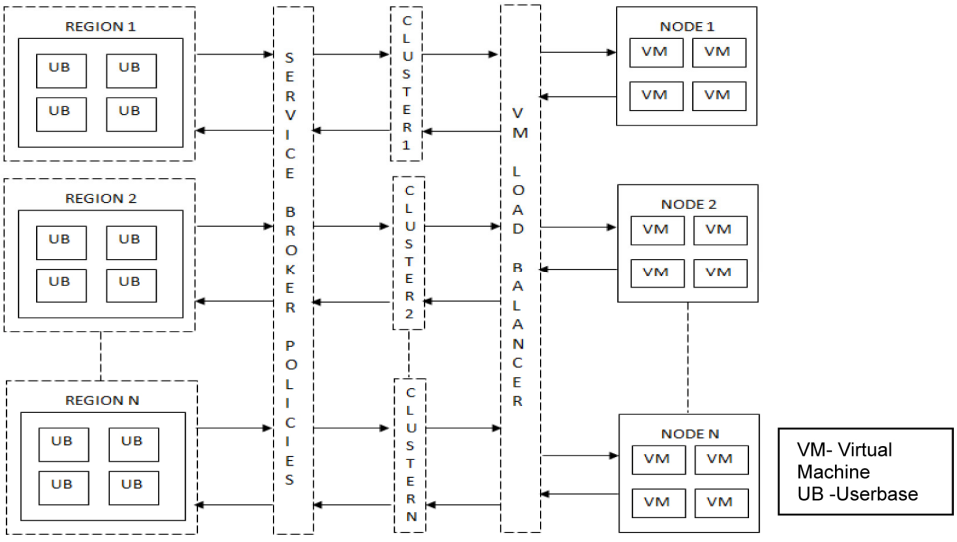


Fig. 2. Generalized architecture of a Cloud

[11]. A Cloudlet is a request received from a Userbase for processing. The VM load balancer plays a very important role in the overall response time of the cloud. Currently, there are three VM Load Balancers that implement three load balancing policies respectively [11] in CloudAnalyst and can be selected as required by the modeler:

1. *Round Robin Load Balancer* – uses a simple Round Robin algorithm to allocate VMs.
2. *Active Monitoring Load Balancer* – this version load balances the tasks between available VMs in a way that evens out the number of active tasks on each VM at any given time.
3. *Throttled Load Balancer* – this ensures that only a pre-defined number of Internet Cloudlets are allocated to a single VM at any given time. If more request groups are present than the number of available VMs at a data center, some of the requests will be queued until the next VM becomes available.

One of the common problem with the above algorithms is that they do not save the state of the previous allocation of a VM to a request from given Userbase. As such, every time a request is received from the same Userbase the algorithm needs to be run again, which increases the total response of the requests. In this paper, we have addressed this problem for the Round Robin VM load balancer.

3. Proposed Algorithm: Round Robin With Server Affinity

The proposed algorithm is an improvement over the Round Robin VM load balancing Algorithm. As discussed in Sections 1 and 2, the Round Robin Algorithm does not save the state of the previous allocation of a VM to a request from given Userbase, while the same state is saved in the proposed algorithm. The simulation of the Round Robin Algorithm has been discussed in Section 5 while the simulation of proposed algorithm (i.e., Round Robin with server affinity) is discussed in Section 6. A comparative analysis of the above two algorithms is discussed in Section 7.

The proposed algorithm (i.e. Round Robin with server affinity) VM load balancer and the related flowchart are given in Figures 3 and 4 respectively

The Round Robin with server affinity VM load balancer maintains two data structures, which are as listed below.

1. Hash map: this stores the entry for the last VM allocated to a request from a given Userbase
2. VM state list: this stores the allocation status (i.e., Busy/Available) of each VM.

```
Round_Robin_With_Server_Affinity  ( )
{
Initialize all of the VMs allocation status to AVAILABLE in the VM state list;
Initialize the hashmap with no entries;
While (new requests are received by the Data center Controller)
do      {
  Data center Controller queues the requests;
  Data center Controller removes a request from the beginning of the queue;
  If (Hashmap contains any entry of a VM corresponding to the currently requesting Userbase &&
  VM allocation status = =AVAILABLE)
      {
      the VM is reallocated to the Userbase request;
      }
  else
      {
      Allocate a VM to the Userbase request using the Round Robin algorithm;
      Update the entry of the Userbase and the VM in the hashmap and the VM state list;
      }
}
```

Fig. 3. The Round Robin with server affinity VM load balancer Algorithm

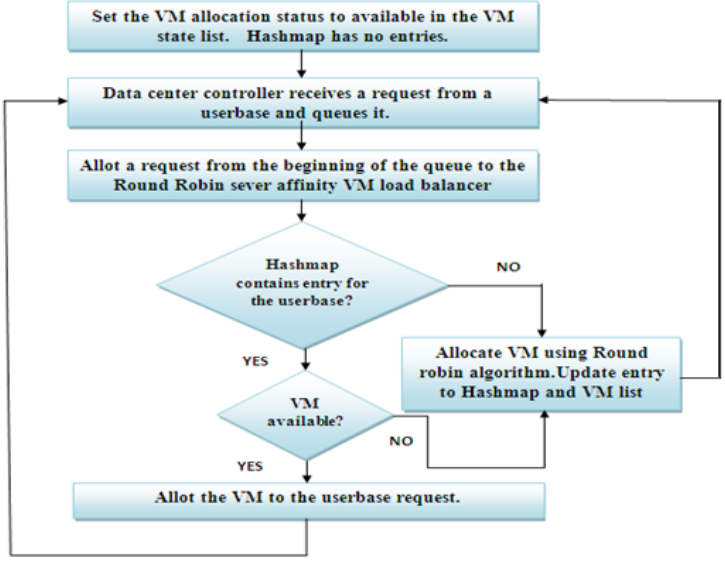


Fig. 4. Flowchart for the Round Robin with server affinity VM load balancer

In the proposed algorithm, when a request is received from the Userbase, if an entry for the given Userbase exists in the hash map and if that particular VM is available, there is no need to run the Round Robin VM load balancing algorithm, which will save a significant amount of time.

4. EXPERIMENTAL SETUP

To analyze and compare the proposed algorithm with the existing algorithm we have used CloudAnalyst as the simulator**.** The approach adopted by CloudAnalyst involved simulating data of one of the most prominent and popular social networking websites (i.e., Facebook). To carry out simulations and analysis of the proposed algorithm the environment setup and the approach adopted were kept in line with the one adopted by CloudAnalyst [11]. The reason for using this is that a social networking site can greatly benefit by moving to cloud and considering the large number of users on Facebook the data available will be the right choice for testing the efficiency of the proposed algorithm. The approximate distribution of the Facebook [10] Userbase around the world is given in the Table 1 (on Dec. 31, 2011).

4.1 Simulation and Parameter Configuration

Considering the resource and budget constraints, we used a subset of the data given in Table 1 for simulation. We further defined 6 Userbases representing the 6 regions with the parameters shown in Table 1.

For all practical purposes, the following reasonable assumptions can be made:

- Each Userbase is contained within a single time zone.
- Most of the users use the application after work for about 1 hour.
- 0.05% of the registered users will be online during the peak time simultaneously and only 1/10 of that number will be online during the off-peak hours.
- Each user makes a new request every 5 minutes when online.

The parameters for the Cloud configuration are defined in Table 2

4.2 Scenarios and Simulation Output

To analyze and compare the proposed algorithm (i.e., the Round Robin Algorithm with server affinity) with the existing algorithm (i.e., the Round Robin Algorithm) we considered the scenarios of multiple data centers ranging from 1 to 15. This is because no significant changes were observed in the overall Userbase response time and the data center processing time on further increasing the number of data centers and the graphs approached a steady state of Userbase response time and Data center processing time. The detailed explaination of 2 scenarios (for 1 and 2 data centers) for the Round Robin Algorithm and the Round Robin Algorithm with Server affinity is given in Sections 5 and 6 respectively.

The simulation results that have been used for comparing the two algorithms are based on the parameters listed below.

A. Userbase response time
B. Data center processing time

Table 1. Userbase parameter list across the regions of the globe

Region	Region ID	Users	Userbase	Peak hrs. (GMT)	Online users during peak hrs.	Online users during off peak hrs.
North America	0	175 million	**UB1**	13:00-15:00	87,500	8,750
South America	1	103 million	**UB2**	15:00-17:00	51,500	5,150
Europe	2	233 million	**UB3**	20:00-22:00	97,500	9,750
Asia	3	195 million	**UB4**	01:00-03:00	116,500	11,650
Africa	4	38 million	**UB5**	21:00-23:00	19,000	1,900
Oceania	5	13 million	**UB6**	09:00-11:00	6,500	650

Table 2. The Data center and VM configuration used for simulation

Parameters	Value used
VM image size	10,000
VM memory	512 MB
VM Bandwidth	1000
Data center – Architecture	X86
Data center – OS	Linux
Data center – VMM	Xen
Data center – Number of Machines	5
Data center – Memory per Machine	1,024 Mb
Data center – Storage per machine	100,000 Mb
Data center – Available BW per Machine	10,000
Data center – Number of processors per machine	3
Data center – Processor speed	100 MIPS
Data center – VM Policy	Time Shared
User Grouping Factor	1,000
Request Grouping Factor	100
Executable Instruction Length	250

The third output of the simulation is cost, which includes the VM cost and the data transfer cost. This parameter remains same for both the algorithms under comparison in the same scenario. Hence, it has not been used as a basis for comparison. In terms of the cost of hosting applications in a Cloud, CloudAnalyst assumes a pricing plan that closely follows the actual pricing plan of Amazon EC2 [14].

5. Simulation of the Existing VM Load Balancing Policy: The Round Robin Algorithm

5.1 Scenario 1: Simple web application hosted on a single data center with Round Robin VM load balancing policy

Like with most real-world web applications, let us assume that the application is deployed in a single location, in Region 0 (North America). Assuming that the application is deployed on a single data center having 25 virtual machines (with 1,024Mb of memory in each VM running on

physical processors capable of speeds of 100 MIPS).

The simulation output for Scenario 1 is shown in Table 3. The response times experienced by each Userbase are depicted graphically in Figure 5(A). The spikes in response times can be seen clearly during the peak period, and it can be observed how the peak loads of one Userbase could affect other Userbases as well. The average, minimum, and maximum response times are shown in Table 3. Since only 1 data center has been considered for the simulation, the average, minimum and the maximum of the data center processing time is the same as the overall average, minimum, and the maximum data center processing time respectively and it is given in Table 3. The graph for the data center processing time for DC1 is shown in Figure 5(B).

Table 3. Simulation output for Scenario 1

A. Userbase Response Time			B. Data Center Processing Time		
Avg. (ms)	**Min (ms)**	**Max (ms)**	**Avg. (ms)**	**Min (ms)**	**Max (ms)**
3,010.80	205.62	9,579.51	2,683.80	15.09	9,082.85

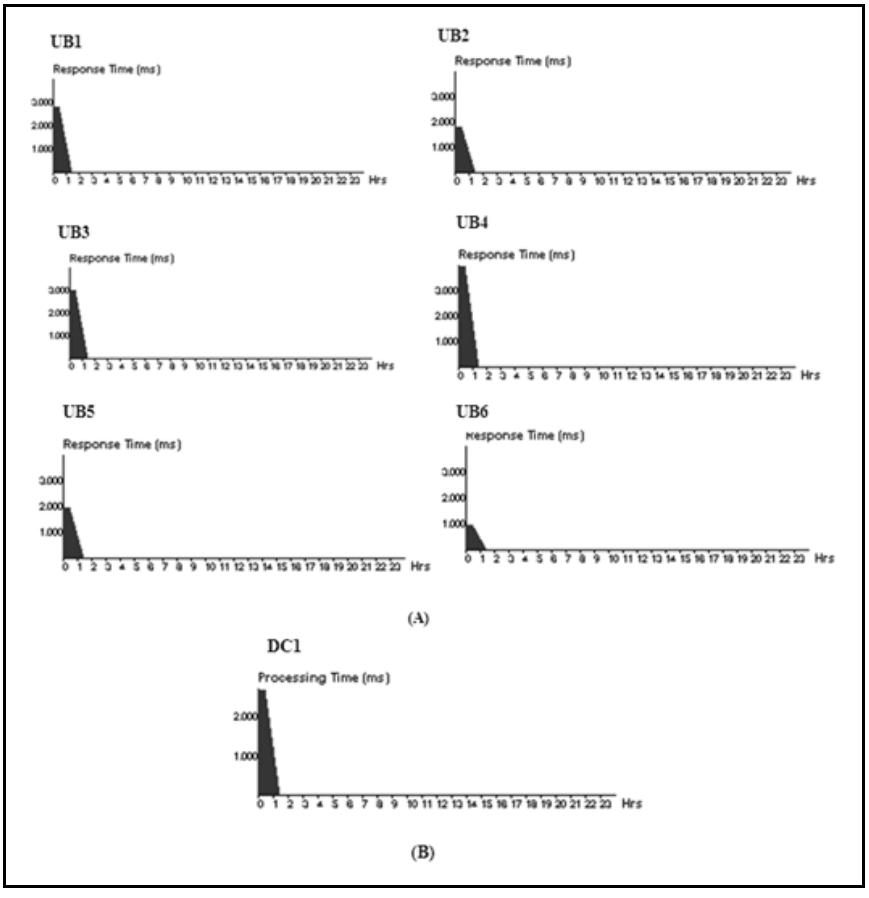


Fig. 5. Scenario 1 (A) Userbase response times experienced by each (B) Data center processing Time

5.2 Scenario 2: Simple web application hosted on multiple data centers with the Round Robin VM load balancing policy

The application is deployed in distributed locations in Region 0 (North America). Assuming that the application is deployed on 2 data centers with 25 virtual machines each (with 1,024 Mb of memory in each VM running on physical processors capable of speeds of 100 MIPS).

The simulation output is shown in Table 4 and Figure 6.

Table 4. Simulation output for Scenario 2

A. Userbase response time			B. Data center processing time			
Avg. (ms)	**Min (ms)**	**Max (ms)**	**Data center**	**Avg. (ms)**	**Min (ms)**	**Max (ms)**
1,764.92	136.07	5,557.98	DC 1	1,479.88	8.11	5,130.79
			DC 2	1,388	20.71	3,883.90
			Overall	1,434.45	8.11	5,130.79

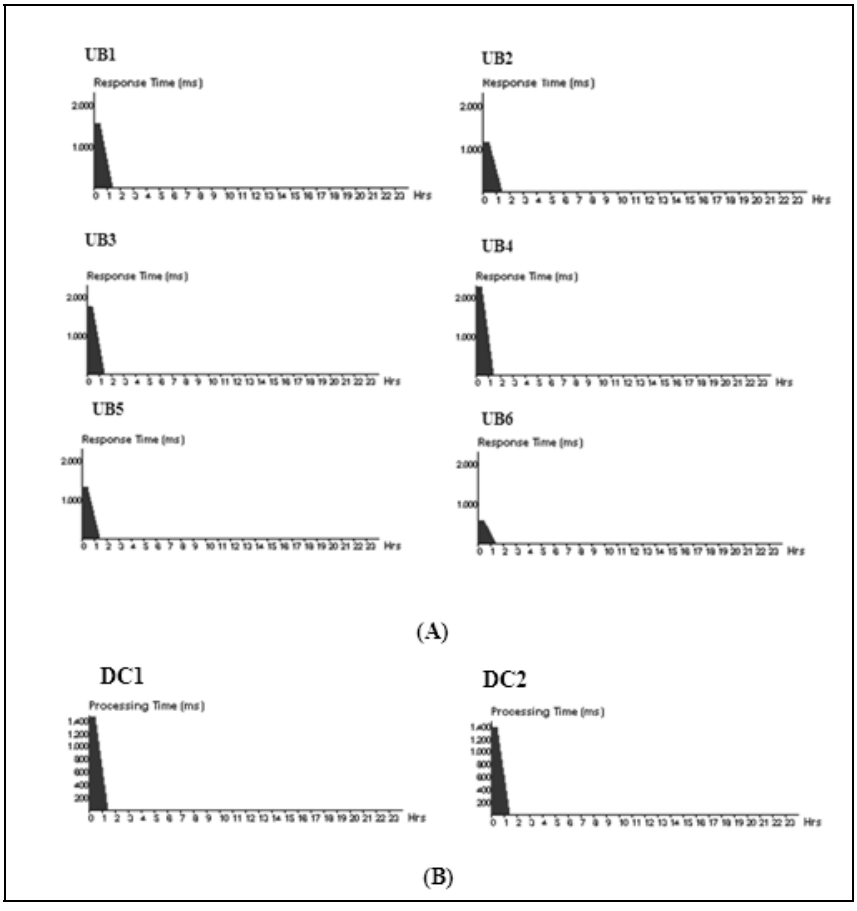


Fig. 6. Scenario 2 (A) Response times experienced by each Userbase (B) Data center processing time

6. Simulation of the Proposed VM Load Balancing Policy: The Round Robin With Server affinity Algorithm

6.1 Scenario 1: Simple Web Application hosted on a single data center with Round Robin with server affinity VM load balancing policy

The scenario assumptions are the same as described in Section 5.1. The simulation output is shown in Table 5 and Figure 7.

Table 5. Simulation output for Scenario 1

Userbase Response Time			Data center Processing Time		
Avg. (ms)	Min (ms)	Max (ms)	Avg. (ms)	Min (ms)	Max (ms)
3,121.58	205.62	9,825.77	2,794.22	15.09	9,302.58

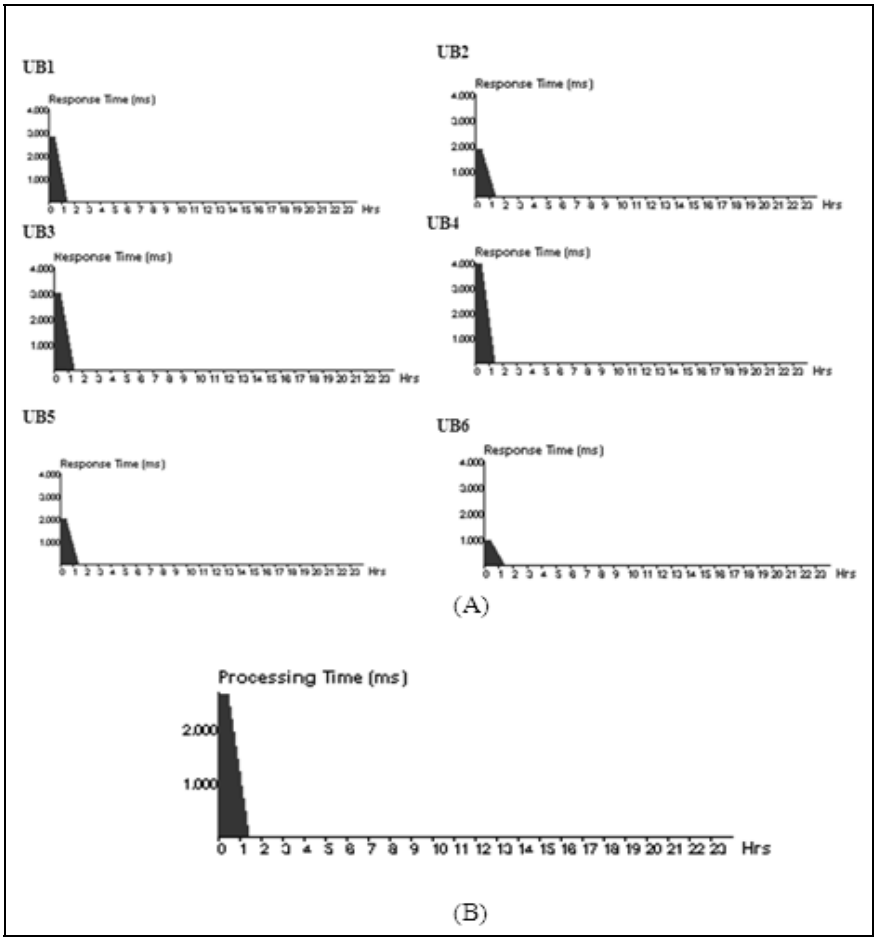


Fig. 7. Scenario 2 (A) Response times experienced by each Userbase (B) Data center processing time

6.2 Scenario 2: Simple Web Application hosted on multiple Data centers with Round Robin with server affinity VM load balancing policy

The scenario assumptions are the same as described in Section 5.2. The simulation output is shown in Table 6 and Figure 8.

Table 6. Simulation output for Scenario 2

A. Userbase Response Time			B. Data center Processing Time			
Avg. (ms)	Min (ms)	Max (ms)	Data center	Avg. (ms)	Min (ms)	Max (ms)
1723.66	193.91	5308.17	D1	1,445.68	20.71	4,794.98
			D2	1,430.33	13.11	4,124.23
			Overall	1,338.12	13.11	4,794.98

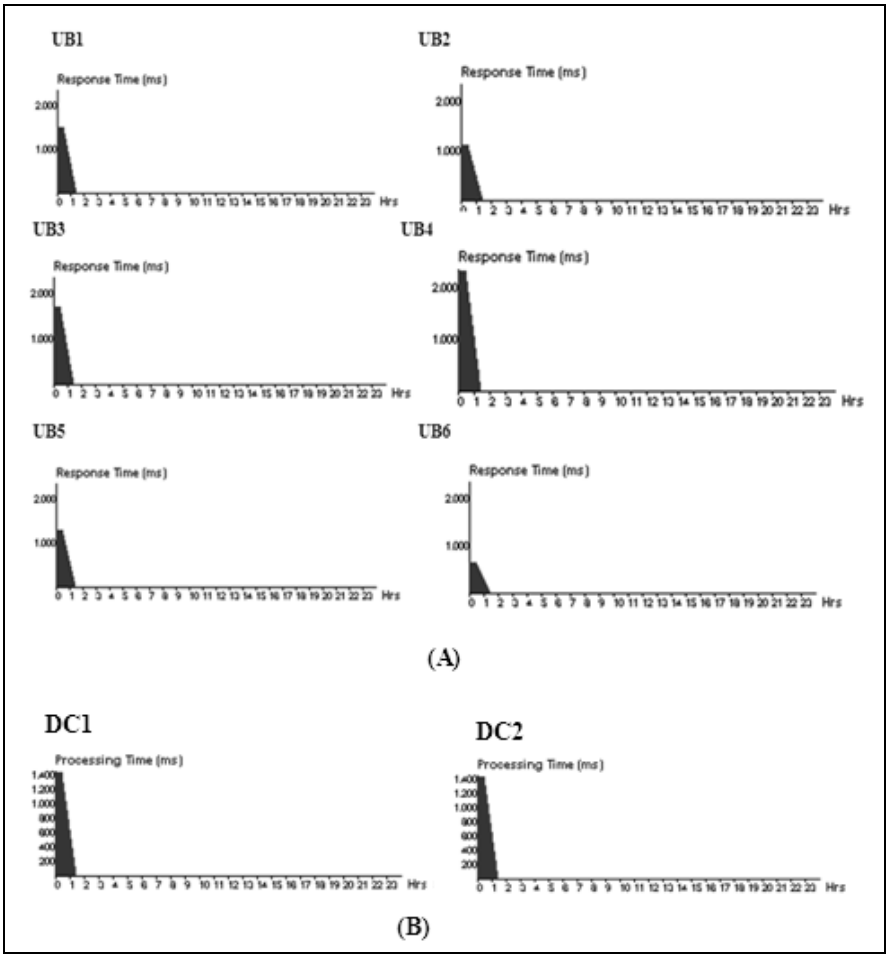


Fig. 8. Scenario 2 (A) Response times experienced by each Userbase (B) Data center processing time

7. COMPARATIVE ALGORITHM ANALYSIS: THE ROUND ROBIN AND THE ROUND ROBIN WITH SERVER AFFINITY VM LOAD BALANCING ALGORITHMS

The comparative analysis summarized from Table 7 and Figures 9 and 10 is given below.

- It is observed that in the case of a single data center with 25 VMs, the overall average response time and the overall average data center processing time increases when we add server affinity to the Round Robin VM load balancing policy.
- It is observed that by increasing the number of data centers, the overall average response time and the overall average time spent for processing a request by a data center decreases in both of the VM load balancing algorithms.
- However, when we compare the overall average response time and the overall average data center processing time for the two algorithms when there is more than one data center, bet-

Table 7. Overall summarized results

Scenario	Overall Average Response Time (in ms)		Overall Average Data center Processing time (in ms)	
	Round Robin VM load balancing algorithm	Round Robin with server affinity VM load balancing algorithm	Round Robin VM load balancing algorithm	Round Robin with server affinity VM load balancing algorithm
1 Data center with 25 VMs each	3,010.80	3,121.58	2,683.80	2,794.22
2 Data centers with 25 VMs each	1,764.92	1,723.66	1,434.45	1,338.12
3 Data centers with 25 VMs each	1,684.56	1,362.49	1,369.81	1,050.96
4 Data centers with 25 VMs each	1,341.37	1,121.23	1,042.95	874.11
5 Data centers with 25 VMs each	973.49	666.83	668.48	568.64
6 Data centers with 25 VMs each	899.67	532.43	595.55	532.78
7 Data centers with 25 VMs each	763.76	490.36	458.44	436.2
8 Data centers with 25 VMs each	748.21	460.11	444.83	401.01
9 Data centers with 25 VMs each	730.88	450.22	429.94	380.98
10 Data centers with 25 VMs each	720.84	438.16	415.96	376.19
11 Data centers with 25 VMs each	713.69	430.69	411.84	370.42
12 Data centers with 25 VMs each	688.14	426.76	385.19	366.36
13 Data centers with 25 VMs each	675.94	421.73	376.42	362.43
14 Data centers with 25 VMs each	671.38	419.19	366.94	356.22
15 Data centers with 25 VMs each	668.2	416.01	335.86	352.48

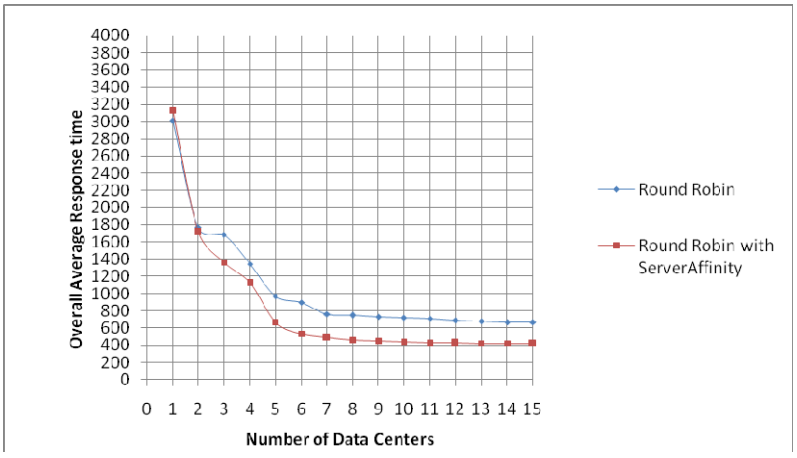


Fig. 9. Graph depicting a comparative analysis of the overall average response time

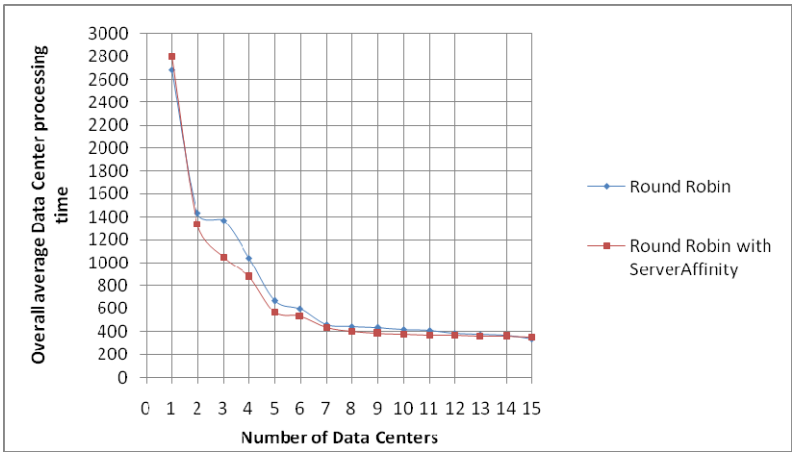


Fig. 10. Graph depicting a comparative analysis of the overall average data center processing time

ter results are observed in the case of the Round Robin with server affinity VM load balancing policy than the Round Robin VM load balancing policy.

- Also, a major decrease in the overall average response time and the overall average data center processing time is noticed up to 6 data centers. However, on further distribution of data centers, no major changes are observed.

To conclude, the proposed VM load balancing algorithm (i.e. Round robin with server affinity) gives better results than the Round robin VM load balancing in algorithm in the case of distributed data centers.

The Table 7 shows the test results for the Round Robin and the Round Robin with server affinity VM load balancing algorithms.

The limitations of the proposed work are described below.

- The algorithm has been tested and analyzed on the simulator, CloudAnalyst. The delays and cost may differ from a real time cloud environment depending on different cloud infrastructures. The simulation results only serve as a basis for comparison.
- Due to budget constraints, the application has been tested on considerably low cost hardware. However, in the future the authors plan to test the application on more powerful hardware.
- The work does not deal with any specific network topology.

8. CONCLUSION & FUTURE WORK

Irrespective of the developments in the IT industry, Cloud computing is an evolving computing paradigm has influenced every other entity in the globalized industry, whether it is in the public sector or the private sector. Considering the growing importance of cloud, finding new ways to improve cloud services is an area of concern and research focus.

There are only a limited number of VM load balancing policies available at present. Also, in the existing VM load balancing strategies, the algorithm has to be run every time a new request for VM allocation is received from a given Userbase. This is because they do not save the state of the previous allocation of a VM to a request from a given Userbase.

The research work involved developing an efficient VM load balancing algorithm for the cloud and conducting a comparative analysis of the proposed algorithm with the existing algorithms. Intermediate deliverables included studying the existing VM load balancing algorithms, proposing an efficient algorithm for VM load balancing, implementing the algorithm on CloudAnalyst, and comparing the proposed algorithm with the existing algorithms on identified parameters.

The major contribution of the proposed work in the field of cloud computing is that the authors have introduced an improved algorithm functionality for an efficient VM load balancing policy that reduces the response time and processing time in the case of distributed data centers.

Our future work will encompass testing the proposed algorithm on a Eucalyptus based private cloud, testing the algorithm on more powerful hardware, exploring more efficient VM load balancing and service load balancing (service brokerage) algorithms, and incorporating failure handling mechanisms into the simulation.

REFERENCES

[1] http://safegov.org/2012/2/16/the-president's-budget-making-cloud-computing-a-priority-for-the-future as on Sep. 2012.

[2] Foster, I; Yong Zhao; Raicu, I.; Lu, S. "Cloud Computing and Grid Computing 360-Degree Compared", published in Grid Computing Environments Workshop, 2008. GCE '08 IEEE DOI 12-16 Nov. 2008.

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

[4] Sun Microsystems, Inc."Introduction to Cloud Computing Architecture" Whitepaper, Ist Edition, June 2009.

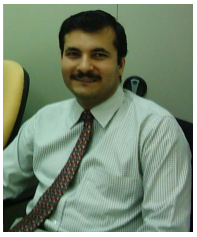
[5] Bhathiya Wickremasinghe, Rodrigo N. Calheiros, and Rajkumar Buyya "CloudAnalyst: A CloudSim-based Visual Modeller for Analysing Cloud Computing Environments and Applications" ; Technical Report, CLOUDS-TR-2009-12, Cloud Computing and Distributed Systems Laboratory, The University of Melbourne, Australia, Oct. 23, 2009.
[6] R. Buyya, R. Ranjan, and R. N. Calheiros, "Modeling and Simulation of Scalable Cloud Computing Environments and the CloudSim Toolkit: Challenges and Opportunities," Proc. of the 7th High Performance Computing and Simulation Conference (HPCS' 09), IEEE Computer Society, June 2009.
[7] http://www.dcs.ed.ac.uk/home/hase/simjava/ as on March 2012.
[8] F. Howell and R. Macnab, "SimJava: a discrete event simulation library for Java," Proc. of the 1st International Conference on Web based Modeling and Simulation, SCS, Jan. 2008.
[9] R. Buyya, and M. Murshed, "GridSim: a toolkit for the modeling and simulation of distributed resource management and scheduling for Grid computing," Concurrency and Computation: Practice and Experience, vol. 14, Nov. 2002, pp. 1175-1220.
[10] http://www.facebook.com as on Dec. 2011.
[11] Bhathiya Wickremasinghe "CloudAnalyst: A CloudSim-based Tool for Modelling and Analysis of Large Scale Cloud Computing Environments "MEDC Project Report.
[12] A. Legrand, L. Marchal, and H. Casanova, "Scheduling distributed applications: the SimGrid simulation framework," Proc. of the 3rd IEEE/ACM International Symposium on Cluster computing and the Grid (CCGrid 07), May 2001, pp. 138-145.
[13] Qi Zhang, Lu Cheng, Raouf Boutaba, Cloud computing: state-of-the-art and research challenges, Journal of Internet Services and Applications, Springer, May 2010, Volume 1, Issue 1, pp 7-18.
[14] http://aws.amazon.com/ec2/ "Amazon Elastic Compute Cloud (Amazon EC2)," as on Sep. 2012

Komal Mahajan

She is presently working as an Assistant Professor in the Department of CSE/ICT at Jaypee University of Information Technology (JUIT), Waknaghat, India and pursing her Doctoral research work in the area of Cloud Computing. She has received her M.S In Software Systems from BITS, Pilani and B.E. in Computer Science and Engineering from MIET.

Ansuyia Makroo

She is presently working as an Assistant Professor in the Department of CSE/ICT at Jaypee University of Information Technology (JUIT), Waknaghat, India and pursing her Doctoral research work in the area of Cloud Computing. She has received her M.S In Software Systems from BITS, Pilani and B.E. in Computer Science and Engineering from MIET.



Deepak Dahiya

He is currently working as Professor in the Department of CSE & ICT at Jaypee University of Information Technology (JUIT), Waknaghat, India. He has M.S. and PhD degrees in Computer Science from BITS Pilani, India. Deepak has over 20 years of experience in IT Industry and Academics in India, Australia, US, UK and Oman. Deepak is a Visiting Researcher to RMIT University, Australia, Guest Faculty to FMS Delhi, IIM Rohtak, India, IIM Kozhikode, India and LNMIIT Jaipur, India. In the IT Industry, he has consulted for Corporate Clients in UK, US and India. He is a senior member of IEEE and life member of Computer Society of India.