Modified Active Monitoring Load Balancing with Cloud Computing

Vikas kumar[1], Shiva Prakash[2]
[1,2]Department of Computer Science & Engineering.
[1,2]Madan Mohan Malaviya University of Technology, Gorakhpur (U.P.), India

Abstract— Cloud computing is internet-based computing in which large groups of remote servers are networked to allow the centralized data storage, and online access to computer services or resources. Load Balancing is essential for efficient operations in distributed environments. As Cloud Computing is growing rapidly and clients are demanding more services and better results, load balancing for the Cloud has become a very interesting and important research area. In the absence of proper load balancing strategy/technique the growth of CC will never go as per predictions. The main focus of this paper is to verify the approach that has been proposed in the model paper [3]. An efficient load balancing algorithm has the ability to reduce the data center processing time, overall response time and to cope with the dynamic changes of cloud computing environments. The traditional load balancing Active Monitoring algorithm has been modified to achieve better data center processing time and overall response time. The algorithm presented in this paper efficiently distributes the requests to all the VMs for their execution, considering the CPU utilization of all VMs.

Key words: Cloud, Virtualization, Load Balancing, Load Balancer

I. Introduction

Cloud computing is an on demand service in which shared resources, information, software and other devices are provided according to the clients requirement at specific time. It’s a term which is generally used in case of Internet as shown in Fig 1. The whole Internet can be viewed as a cloud. Capital and operational costs can be cut using cloud computing.

Load balancing in cloud computing systems is really a challenge now. Always a distributed solution is required, because it is not always practically feasible or cost efficient to maintain one or more idle services just as to fulfill the required demands. Jobs can’t be assigned to appropriate servers and clients individually for efficient load balancing as cloud is a very complex structure and components are present throughout wide spread area. Here some uncertainty is attached while jobs are assigned.

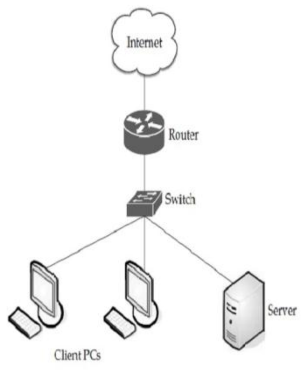


Figure 1: A cloud is used in network diagrams to depict the Internet (adopted from [1]).

This paper considers some of the methods of load balancing in large scale Cloud systems. Our aim is to provide an evaluation and comparative study of the proposed approach and to improve the different performance parameters like data processing time, response time etc. for the clouds of different sizes. As the whole Internet can be viewed as a cloud of many connection-less and connection oriented services, thus concept of load balancing in Wireless sensor networks (WSN) proposed in [2] can also be applied to cloud computing systems as WSN is analogous to a cloud having no. of master computers (Servers) and no. of slave computers (Clients) joined in a complex structure. A comparative study of proposed algorithms and VM assign algorithm has been carried.

The rest of the paper is organized as follows: Section II, focuses on various load balancing strategies based on which several algorithms have been suggested. Section III, discusses about the existing load balancing algorithms as a literature review. Section IV, represents previous work, the problem associated with the previous work and the proposed algorithm that efficiently balances the load. Section V shows the experimental setup under which the proposed algorithm has been implemented. Section VI shows the results illustrating how the proposed algorithm produces better Data center processing time and overall response time and finally Section VII concludes the paper with future scope.

II. Load Balancing

A. Load Balancing in Cloud Computing (CC)

It is a process of reassigning the total load to the individual nodes of the collective system to make resource utilization effective and to improve the response time of the job, simultaneously removing a condition in which some of the nodes are over loaded while some others are under loaded. A load balancing algorithm which is dynamic in nature does not consider the previous state or behavior of the system, that is, it depends on the present behavior of the system. The important things to consider while developing such algorithm are : estimation of load, comparison of load, stability of different system, performance of system, interaction between the nodes, nature of work to be transferred, selecting of nodes and many other ones [4] . This load considered can be in terms of CPU load, amount of memory used, delay or Network load.

Depending on who initiated the process, load balancing algorithms can be of three categories as given in [4]:

- Sender Initiated: If the load balancing algorithm is initialized by the sender
- Receiver Initiated: If the load balancing algorithm is initiated by the receiver
- Symmetric: It is the combination of both sender initiated and receiver initiated

Depending on the current state of the system, load balancing algorithms can be divided into 2 categories as given in [4]:

1) Static

Static load balancing is the simplest form of the two types. Static load balancing is the selection and placement of a logical process on some processing element located on a distributed network. The selection of the processing element for some logical process is based upon some weighting factor for that process, or some kind of workload characterization of the processing elements or of the network topology, possibly both. The process of selecting a processing element for a logical process requesting service may be done with two possible methods, a stateless method or a state-based method. With a stateless method the selection of a processing element is done without regard to any knowledge of the system state. A state-based method of selecting a processing element implies that the selection of a processing element requires knowledge of the system state, either globally, or locally. Global knowledge implies that the state of the all the components of the system is known and local knowledge implies that only partial knowledge is known. If the state of the system is needed then some kind of messaging or probing of network resources among the requesting processes, agents, or processing elements is needed to determine their availability. Once a processing element is selected for a logical process, the logical process is executed on the selected processing element for the duration of the logical process lifetime.

B. Dynamic

Decisions on load balancing are based on current state of the system. No prior knowledge is needed. So it is better than static approach. Dynamic load balancing is the initial selection and placement of a logical process on some processing element in a distributed network and then at some point in time there may be a decision made to move a process to some other processing element. The initial selection and placement is done with the same method as static load balancing. But at some point in time during the execution of the logical process, based on some decision criteria, a process may be preempted and migrated to another processing element somewhere else on the network. Generally a process is migrated to another processor if the migration cost or overhead is less than some predetermined metric.

III. LITERATURE REVIEW

In this section we discuss the most known contributions in the literature for load balancing in Cloud Computing.

Jasmin James et al. [10] analyzed various VM load balancing algorithms. Then, the author has proposed a new load balancing algorithm and implemented it for an IaaS framework in Simulated cloud computing environment i.e.”Weighted Active Monitoring Load Balancing Algorithm”.

In the VM load balancing algorithm the author has proposed an approach in which individual application services is assigned varying (different) amount of the available processing power of VMs This is because- in the real world, it’s not necessary all the VMs in a DataCenter has fixed amount of processing powers but it can vary with different computing nodes at different ends. And then to these VMs of different processing powers, the tasks/requests (application services) are assigned or allocated to the most powerful VM and then to the lowest and so on. They are given the required priority weights. Hence, the performance parameters such as overall response time and data processing time are optimized.

The author has concluded that the proposed load balancing algorithm has optimized the given performance parameters such as response time and data processing time, giving an efficient VM load balancing algorithm.

Jaspreet Kaur et al. [11] have defined the problem as, the random arrival of load in such a environment can cause some server to be heavily loaded while other server is idle or only lightly loaded .Equally load distributing improves performance by transferring load from heavily loaded server. Thus, the author has provided the solution based on scheduling algorithm ECSE(Equally Spread Current Execution)load. The author has then compared the performance result of ECSE with round robin scheduling to estimate response time, processing time.

In the algorithm proposed by the author the active VM load balancer finds the next available VM. The load balancer checks that for all current allocation count is less than maximum length of VM list in order to allocate the VM. If available VM is not allocated then a new one is created. And at last active load is counted on each VM and id of those VM is returned which is having least load.

The author concludes his paper after simulating the algorithm on CloudSim. The ESCE algorithm dynamically allocates the resources to the job in queue leading reduced cost in data transfer and virtual machine formation. The simulation result shows overall time cost results and comparison of load balancing algorithms. ESCE load balancing provide better results as compared to round robin on closet data center, optimize response time.

Nyandeep Sran et al. [12] developed a Load Balancer Algorithm that controls the flow of payload based on the safety thresholds, which may be static or dynamic in nature, depending on the available machines and bandwidth as well. The author has analyzed the existing algorithms of Load Balancing such as Round Robin, Throttled, Equally Spread and Biased Random Sampling and has proposed a new algorithm which will meliorate the existing Load Balancing Approach, by decreasing the overall requesting time and processing time as compared to the existing algorithms and hence will decrease the cost which is proved through rigorous simulation study. The Proposed Algorithm will also provide security to the data in cloud during Load Balancing process by using Zero Proof Algorithm.

In the proposed approach the author is working on VM migration policy by giving priority to VM’s on the basis of resources available to it. Firstly, the algorithm will check that whether the CPU utilization of VM(Virtual Machine) is equal to, greater than or less than 80%. The value has been chosen in order to prevent the machine from being overloaded.

It is clear from the results that the proposed method in the paper is able to balance the load to a greater extent than the analyzed algorithms. A comparative study is done in this thesis with the Proposed Algorithm. This Load Balancing algorithm aims at providing dynamic, on-demand, balance of resources available to the resources required to accomplish the task. Since, this algorithm involves reallocation of resources involving VM and data centers. Therefore, every time whenever reallocation occur re-authentication of the resource allocation is also conducted along with reallocation of Load Balancing. So, this algorithm also provides better security, by focusing on the concept of not disclosing the personal details of the user to cloud provider.

Anjali D.Meshram et al. [13] proposed that the scheduling strategy should be developed for multiple tasks. In cloud computing processing is done on remote computer hence there are more chances of errors, due to the undetermined latency and lose control over computing node. The author of the paper solves this determines this problem and tries to give a fault tolerant model for the cloud computing. The FTMC model tolerates the faults on the basis of reliability of each computing node. A Computing node is selected for computation on the basis of its reliability and can be removed, if does not perform well for applications.

The approach used by the author in this paper: Reliability assessment algorithm is applied on each node (virtual machine) one by one. Initially reliability of a node is set to 1. There is an adaptability factor ***n***, which controls the of reliability assessment. The value of *n* is always greater than 0. The algorithm takes input of three factors *RF*, minReliability and maxReliability from configuration file. *RF* is a reliability factor which increases or decreases the reliability of the node. Min Reliability is the minimum reliability level. If a node reaches to this level, it is stopped to perform further operations.

The author concludes the paper with the simulation results showing that the proposed fault tolerant technique works better than other similar fault tolerant techniques.

IV. PROPOSED WORK

A. Problem Statement

In the paper [6], the author G.Domanal et al. have proposed the approach of load balancing by modifying the active load balancing strategy. The authors have added a condition to check whether the chosen VM and last used VM are same or not. If they are not the same, then the request is assigned to the VM, otherwise the procedure is repeated.

The proposed strategy of the author does not work well with heterogeneous configuration of the VMs. But, in our proposed approach we do not consider the number of requests running on the VMs to decide for the VM assignment, instead we consider the CPU utilization. Since, the number of processors and other specifications of the VMs vary, so we focus on the CPU utilization of all VMs for the assignment.

B. Proposed Strategy

The CPU utilization of the VMs can be obtained through VMs itself. There are several tools like vSphere, Cisco IOS Software-Based Tool, OpUtils which can calculate the CPU utilization. We need to install the tool on the VM and then record the CPU utilization for every VM. Another way to calculate the CPU utilization is in terms of different parameters like bandwidth, RAM and MIPS. In our implementation we have used the basic formula for the calculation of CPU utilization. We know that calculating the utilization is not an easy task.

For our purpose, we define CPU utilization, U, as the amount of time not in the idle task, as shown in Equation 1.

U=100% - (% of time spent in the idle task)

The idle task is the task with the absolute lowest priority in a multitasking system. This task is also sometimes called the background task or background loop. The percentage time in the idle task can be calculated as:

%time in = avg. period of background task with no load * 100
idle task avg. period of background task with load

Now, we use the calculated CPU utilization to find the least loaded VM.

When a request arrives the DCC, it asks the load balancer for the VM id for the request allocation. The load balancer checks the index table to find out the VM with least CPU utilization. The id of the VM with minimum CPU utilization is sent to the DCC by the load balancer. The DCC allocates the VM for the incoming request and notifies the load balancer so that it can update the table. The VMs periodically send their CPU utilization to the load balancer which is updated in the table maintained by the Load Balancer.

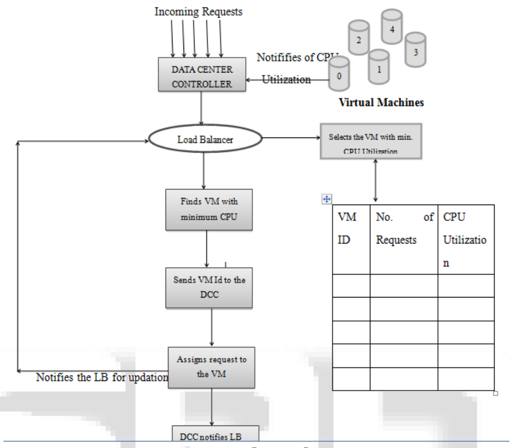

Incoming Requests
DATA CENTER CONTROLLER
Notififies of CPU Utilization
2
4
3
1
0
Virtual Machines
Load Balancer
Selects the VM with min. CPU Utilization
Finds VM with minimum CPU
VM ID
No. of Requests
CPU Utilization
Sends VM Id to the DCC
Assigns request to the VM
Notifies the LB for updation
DCC notifies LB

Fig.2: Flowchart

The figure 2. shows the flowchart of the proposed algorithm. It shows the series of steps involved in the algorithm.

C. Proposed Algorithm

Input : Number of incoming jobs x_1, x_2, x_3.......x_n
Available VM y_1, y_2, y_3.......y_n
Output : All incoming jobs $x_{1,}$ $x_{2,}$.......x_n are allocated least loaded virtual machines among the available $y_{1,}$ y_2.........y_n

- Initially all the VMs have 0 allocations.
- VM- assign load balancer maintains an index/ assign table of VMs which has number of requests currently allocated and the CPU Utilization of each VM.
- When a request arrives the data canter, the DCC parses the load balancer for the appropriate VM.
- The LB checks for the least loaded VM in the index table, by selecting the VM with minimum CPU utilization.
- The LB notifies the DCC about the selected VM by sending the VM id.
- Request is assigned to the VM. DCC notifies the VM- assign load balancer about the allocation.
- The VMs send their CPU utilization to the DCC for the index table periodically.
- VM- assign load balancer updates the requests hold by each VM.
- When VM finishes processing the request , data center receives the response.
- DCC notifies VM- assign load balancer for the VM de-allocation and VM assign load balancer updates

the table.

– Repeat from step 2 for next request.

V. EXPERIMENTAL SETUP

The algorithm proposed in this paper has been implemented on the Cloud Analyst simulator which is built directly on top of cloudsim framework.

Cloud Analyst simulator gives the real time scenario with six different geographical locations. i.e no of users from particular locations can be identified depending on the specific application, e.g. Facebook users from Asia, Africa etc. The simulator is very flexible and it provides data centers, virtual machines, band width and many more for experimentation [7]. A snapshot of the Cloud Analyst architecture is shown in Fig. 3

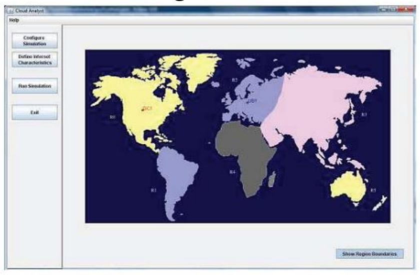


Fig.3: Simulation Screen

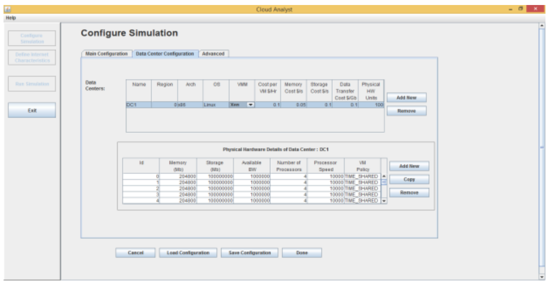
Hypothetical applications like Facebook users, Twitter users, Internet users are considered for experimentation. Six different geographical locations (six different continents of the world) are considered [8]. A single time zone is considered for all user locations. For simplicity one hundredth of the total users from each continent is considered and it is assumed that only 5% of total users are online during peak hours and in off peak hours, users are one tenth of the peak hours.

For experimentation internet users at six different continents are considered i.e. six user bases and peak and non peak users are given in the table 1. We have considered internet users at different continents from the month of June 2012. The same data is experimented with different load balancing algorithm [6] and response time of each algorithm is also considered for the result analyses.

User Base	Simultaneous online Users during peak Hours	Simultaneous online Users during peak Off Hours
UB1	10000	1000
UB2	10000	1000
UB3	10000	1000
UB4	10000	1000
UB5	10000	1000
UB6	10000	1000

Table.1: User base Table

– **Configure Simulation -Data Center Configuration**



Cloud Analyst

Configure Simulation

Main Configuration | Data Center Configuration | Advanced

Data Centers:

Name	Region	Arch	OS	VMM	Cost per VM $/Hr	Memory Cost $/s	Storage Cost $/s	Data Transfer Cost $/Gb	Physical HW Units
DC1	0	x86	Linux	Xen	0.1	0.05	0.1	0.1	100

Physical Hardware Details of Data Center : DC1

Id	Memory (Mb)	Storage (Mb)	Available BW	Number of Processors	Processor Speed	VM Policy
0	204800	100000000	1000000	4	10000	TIME_SHARED
1	204800	100000000	1000000	4	10000	TIME_SHARED
2	204800	100000000	1000000	4	10000	TIME_SHARED
3	204800	100000000	1000000	4	10000	TIME_SHARED
4	204800	100000000	1000000	4	10000	TIME_SHARED

Fig.4: Configuring the simulation

– **Simulation Screen after the completion of Simulation**

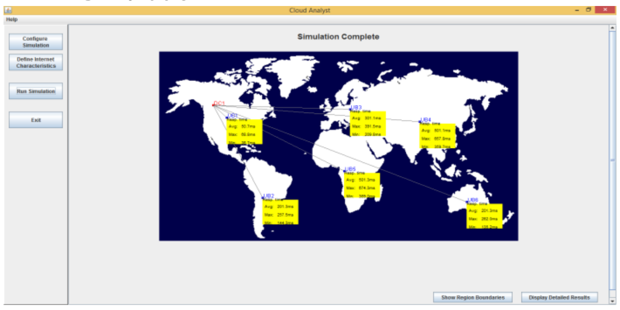


Fig. 5: Simulation Screen after simulation

VI. SIMULATION RESULT& ANALYSIS

In this section the results obtained after simulating the above scenario are discussed.

The first screen after the simulation is shown in fig.6.

Fig. 6: Simulation Screen

Simulation scenario was setup as explained in the previous heading. We have considered a single Data Center (DC) with 100 VMs. Simulation was repeated for proposed Modified Active Monitoring algorithm and VM-assign algorithm [6] respectively. With the proposed algorithm response time for request has been improved compared to existing algorithm proposed in [6]. The graph is plotted with Data Center Processing time in x-axis and overall response time in y-axis. In the fig.6 the graph shows that the performance of our proposed algorithm is better than the existing algorithm.

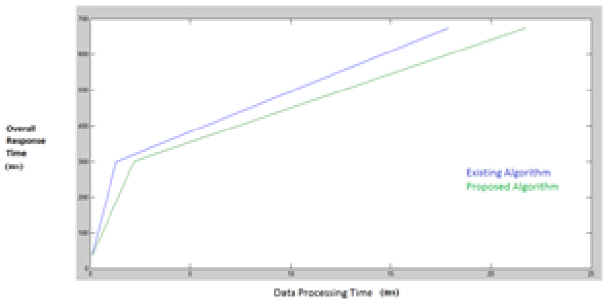


Fig. 7: Comparison of Data Center Processing Time

The following table 2 gives the information about average response time and Data Center processing time of the proposed algorithm.

	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	301.37	36.82	674.39
Data Center processing time:	2.22	0.11	21.75

Proposed Algorithm

	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	300.48	36.72	674.29
Data Center processing time:	1.32	0.10	17.91

A. *Response time of User Bases*

The graph is plotted with User Bases in x-axis and average response time in y-axis. In the fig. 8 a graph of response time vs user bases is shown.

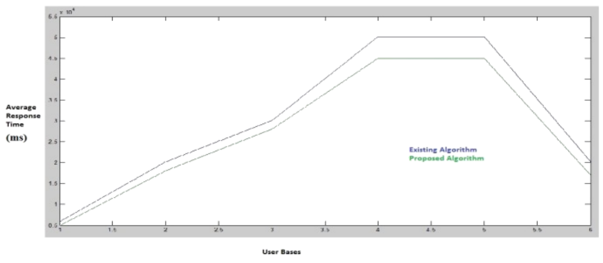


Fig. 8: Comparison of Data Center Processing Time

The performance of our proposed approach is better as compared to the previous algorithm, due to the better response time achieved.

1) *Existing Algorithm*

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	58.66	36.82	79.21
UB2	201.71	144.38	258.46
UB3	301.39	209.92	391.58
UB4	501.31	359.78	663.47
UB5	501.58	369.08	674.39
UB6	201.67	135.32	262.07

2) *Proposed algorithm*

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	54.58	36.72	77.29
UB2	201.45	144.28	258.03
UB3	301.22	209.82	391.48
UB4	501.14	359.68	657.81
UB5	501.40	368.98	674.29
UB6	201.40	135.22	261.97

Table 3: Response Rime by all User bases

B. *Data Center request servicing time*

The graph is plotted with minimum, average and maximum in x-axis and Data Center request servicing time in y-axis.

The fig. 9 shows the bar graph of our proposed algorithm performing better than the existing approach by providing reduced Data center request servicing time

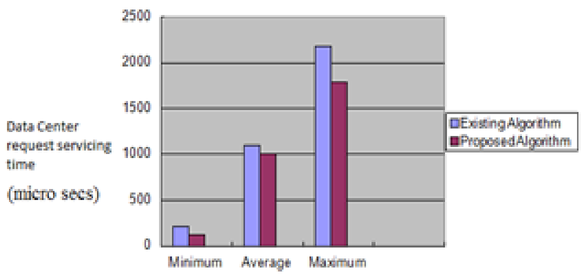


Fig. 9: Comparison of Data Center Processing Time

Now we the improvement in Data Center Request Servicing Time in our proposed algorithm as compared to previous algorithm can be seen in the following tables.

1) *Existing Algorithm*

Data Center	Avg (ms)	Min (ms)	Max (ms)
DC1	2.22	0.11	21.75

2) *Proposed algorithm*

Data Center	Avg (ms)	Min (ms)	Max (ms)
DC1	1.32	0.10	17.91

VII. CONCLUSION

In our research work an efficient algorithm is designed which manages the load at the Datacenters by considering the current status of the all available VMs for assigning the incoming requests intelligently. The Modified active monitoring load balancer mainly focuses on the efficient utilization of the resources i.e VMs. We proved that our proposed algorithm optimally distributes the load and hence under or over utilization (VMs) situations will not arise. When compared to existing VM assign load balance algorithm, the load was not efficiently distributed on the VMs and we can say that the hourly average .processing time of the data center of the proposed algorithm is better. It proves that the request servicing time and the response time of our algorithm is better. Our proposed algorithm solves the problem of inefficient utilization of the VMs resources compared to existing algorithm.

As a future scope the proposed algorithm can still be improved by taking some more dynamic situations of the incoming requests and how the algorithm responses if we mix both static and dynamic loads. Instead of CPU Utilization some other parameters can be considered for VM allocation or parameters like bandwidth, memory or latency can be used to obtain the utilization.

REFERENCES

[1] Vogels, “A head in the clouds the power of infrastructureas as service”, Proceedings of the 1st Workshop on Cloud Computing and Applications, CCA‟08.

[2] Armbrust, A.Fox, R.Griffith, A.Joseph, R.Katz, A.Konwinski, G.Lee,DPatterson, A.Rabkin, I.Stoica, , “Above the clouds : A Berkeley view of cloud computing, Technical Report UCB / EECS -2009-28,2009.

[3] Vikas Kumar, Shiva Prakash, “Modified Active Monitoring Load Balancing Algorithm in Cloud Computing Environment ”, in International Journal for Scientific Research and Development (IJSRD), E-ISSN 2321-0613, 1879-2158, Vol.2, Issue 5 ,(August 2014).

[4] Rimal B.P., Choi E. and Lumb I., “A Taxonomy and Survey of loud Computing Systems”, IEEE 5th International Joint Conference on INC, IMS and IDC, pp - 44-51, 2009.

[5] Mata-Toledo R. and Gupta P. (2010), “Cloud Load Balancing Techniques: A Step towards Green” Journal of Technology Research, Volume No-2 issue - 1, 1-8.

[6] Shridhar G.Damanal and G. Ram Mahana Reddy, “Optimal Load Balancing in Cloud Computing By Efficient Utilization of Virtual Machines ”Department of Information Technology National Institute of Technology Karnataka Surathkal, Mangalore, India, 978-1-4799-3635-9/14/$31.00 ©2014 IEEE.

[7] B.Wickremasinghe, R.N.Calheiros,R.Buyya, Cloudanalyst: A cloudsim based visual modeller for analysing cloud computing in: Proceedings of the 24th International Conference on Advanced Information Networking and Applications (AINA 2010), Perth, Australia,, 2010.

[8] M.Satyanarayanan, P.Bahl, R.Caceres, N.Davies, “The case for VM-based cloudlets in mobile computing, IEEE Pervasive Computing, volume – 8, 2009, pp - 14–23.

[9] Mei,W., Chan,T. Tse,A , “Tale of clouds : paradigm comparis on sandsome thoughtson research issues”, ProceedingsoftheAsia-PacificServices Computing Conference, APSCC‟ IEEE, 2008, pp - 464–469.

[10] Jasmin James, Dr.Bhupendra Verma “EFFICIENT VM LOAD BALANCING ALGORITHM FOR A CLOUD COMPUTING ENVIRONMENT” International Journal on Computer Science and Engineering (IJCSE), September 2012.

[11] Jaspreet Kaur et al. “Comparison of load balancing algorithms in a Cloud”, International Journal of Engineering Research and Applications (IJERA), May-Jun2012.

[12] Nyandeep Sran,Naveep Kaur et al. “Zero Proof Authentication and EfficientLoad Balancing Algorithm for Dynamic Cloud Environment”,International Journal of Advanced Research in Computer Science and Software Engineering “, 7 July 2013..

[13] Anjali D.Meshram A.S.Sambare et al.” Fault Tolerance Model for Reliable Cloud Computing”, International Journal on Recent and Innovation Trends in Computing and Communication, July 2013.

[14] M. Randles, D. Lamb, and A. Taleb-Bendiab, “A Comparative Study into Distributed Load Balancing Algorithms for Cloud Computing,” 2010 IEEE 24th International Conference on Advanced Information Networking and Applications Workshops, 2010, pp. 551–556.

[15] H.-C. Hsiao, H. Liao, S.-S. Chen, and K.-C. Huang, “Load Balancewith Imperfect Information in Structured Peer-to-Peer Systems,”IEEE Trans. Parallel Distributed Systems,vol. 22, no. 4, pp. 634-649,Apr. 2011.

[16] M. Jelasity, A. Montresor, and O. Babaoglu, “Gossip-Based Aggregation in Large Dynamic Networks,”ACM Trans. Computer Systems,vol. 23, no. 3, pp. 219-252, Aug. 2005.

[17] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, and S. Lu, “BCube: A High Performance, Server-Centric Network Architecture for Modular Data Centers,”Proc. ACM SIGCOMM’ 09, pp. 63-74, Aug. 2009.

[18] Braun, Tracy D, et al. "A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems" Journal of Parallel and Distributed computing , Volume 61, Issue 6, Pages 810 – 837, 2001.

[19] Elzeki, O. M., M. Z. Reshad, and M. A. Elsoud. "Improved Max-Min Algorithm in Cloud Computing", International Journal of Computer Applications, Volume 50, Issue 12, Pages 22-27, 2012 .

[20] Opennebula. (2012, July). The cloud data center management solution. [Online]. Available: http://www.opennebula.org/.