International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 4 Issue 07, July-2015

An Efficient Method for Resource Monitoring in Cloud Nodes using Dynamic Load Balancing Algorithm

Mukram B. Ansari
Department of Computer Engineering
G H Raisoni, Collge of Engineering & Management
Wagholi, Pune, India

Prof. Patil Sachin
Department of Computer Engineering
G H Raisoni, Collge of Engineering & Management
Wagholi, Pune, India

***Abstract*—This Cloud computing is the next invention of computation. Almost all the data required to human being is available on the cloud. Cloud computing is used to provide the resources to client on demand. The resources can be either software or hardware resources. Cloud computing architectures are distributed, parallel and supply the needs of multiple clients in different situation. This distributed architecture deploys resources distributive to deliver services effectively to users in different geographical channels. Clients in a distributed environment generate request at random in any processor. So the major drawback of this randomness is related with task assignment. The imbalanced task assignment to the processor creates inequality i.e., few of the processors are overloaded and few of them are under loaded. The objective of load balancing is to shift the load from overloaded process to under loaded process clearly. Load balancing is one of the vital issues in cloud computing. To achieve high performance, minimum response time and high resource utilization ratio it is required to transfer the tasks between nodes in cloud network. This paper provides information about cloud computing, load balancing methods and the proposed load balancing system for above defined problem.**

***Keywords*—Component; Cloud Computing; Load Balancing; IaaS; Load Balancing Algorithms;PaaS; SaaS**

I. INTRODUCTION

We can define cloud computing as collection of distributed servers which provides services on demand [8]. The services may be software or hardware resources as client requires. There are three major components of cloud computing and these are client, data center and distributed server. The components are used as the end user who interacts with client to avail the services of cloud. The initial component i.e. client would be mobile devices. Second component is data center which is nothing but collection of servers hosting multiple applications [9]. Recently virtualization [6] [7] is used to install software that allows different instances of virtual server applications. The third component of cloud is distributed servers which is present throughout the Internet hosting and used for various applications. But when the user uses this application from cloud, user experiences that he is using this application from its own machine.

Cloud computing supports three types [5] of services like as Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS). SaaS gives software support to client which is not required to install on clients machine. PaaS is used to develop the platform to build an application like database. IaaS is used to provide the computational power to the clients for executing the various tasks from another node.

II. LOAD BALANCING

In cloud system it is possible that some nodes to be overloaded and some of them may be under loaded [9]. This scenario results in low performance. The objective of load balancing is distributing the load in between various nodes in cloud environment. So the major issues in cloud computing is load balancing. [6].

For proper resource utilization, load in the cloud system must be balanced [9] equally. Thus, a load balancing algorithm [1][2][4] helps to balance the total system load by transferring the workload from fully loaded nodes to less loaded nodes. Due to this it's possible to achieve better performance. The response time of processes and total system throughput are the some of the metrics considered during design of whole system [2][3].

The mechanism of load balancing is used to improve the performance of the system and for proper allocation of resources in cloud computing. The various functions ofs of load balancing are [1][5][18]:

- Distribution of load uniformly across all the nodes.
- Getting better the overall performance of the system.
- To minimise response time.
- To gain resource utilization ratio.

This can be explained with a suitable example:

Let us assume that we have developed one popular application and deployed it on cloud for an user. Thousands of users started to use this application. Consider that hundreds of users are using this application at the same time from single machine without load balancing approach. Due to this the specific server may be very busy to execute the user's tasks as per their demands. At the same time the other servers are under loaded or idle mode. Because of the low response and poor performance of the system the user's satisfaction cannot be achieved. [19][20]

This situation can be avoided by applying load balancing concept. Due to load balancing it's possible to distribute user's tasks to other nodes. In this way we can be able to achieve high performance and better response time.

A. Classification of Load-Balancing Algorithms

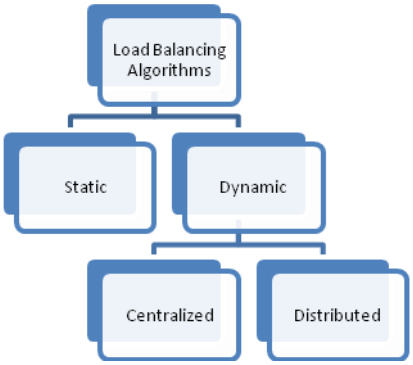


Fig. 1. Classification of Load balancing algorithm

There are main two categories of load balancing is catergoriesd as i) Static load balancing and ii) Dynamic load balancing. [3][4]
Static algorithms do not consider the current state of nodes and works statically whereas Dynamic algorithms [4] operates on present state of node and distributes load in between the nodes. Static algorithms provide the information of average behavior of the system neglecting the current state of system. Dynamic algorithms react to the system state which changes dynamically.
Static load balancing [4] algorithms are easy to use. There is no requirement to maintain and process system state information. However, the weakness of static algorithm is that they do not react to the current system state whereas dynamic algorithms are better as they avoid those states with unnecessarily poor performance. Because of this reason, dynamic policies provide significantly better performance benefits than static [5].

III. LITERATURE SURVEY

The work proposed by many researchers based on load balancing related with cloud computing is summarized below.

A genetic algorithm approach for optimizing the CMSdynMLB was proposed and implemented [1][11]. In their proposed work they presented a model for a practical multiservice dynamic Scenario. This model provides facility in which at different times, clients can change their locations. In this each server cluster handled a specific type of multimedia task which helps to optimize the two performance objectives at the same time. This paper gives information about a mathematical formulation of the CMS-dynMLB problem and also a theoretical analysis for the algorithm convergence.

In the proposed work the authors elaborated the concept of the delay problem on dynamic load balancing for Distributed Virtual Environments (DVEs). Due to communication delays among servers, the load balancing process may be utilizing outdated load information from local servers to calculate the balancing flows. The local Servers would significantly affect the performance of the load balancing algorithm because they may be using outdated balancing flows to do load migration. To overcome this problem two methods are presented: uniform adjustment scheme and adaptive adjustment scheme. The uniform adjustment method performs a uniform distribution of the load variation among the neighbor servers, which is a coarse approximation. The first method is very simple to implement. The adaptive adjustment scheme performs limited degree of user tracking without communicating with neighbor servers [12][13].

In.[14][15], authors have proposed a load balancing technique for cloud computing environments predicated on behavior of honey bee foraging strategy. The proposed algorithm is used to balances the load and also considers the priorities of tasks that have been abstracted from overloaded Virtual Machines. The tasks distracted from these VMs are treated as honey bees. In this approach of Honey bee deportment supports overall throughput of processing and priority predicated balancing fixates by minimizing the duration a task. It has been done by using wait operation on a queue of the VM. Thus, it finds useful in reduction of the replication of time of VMs.
This load balancing technique provides better solution for heterogeneous cloud computing systems and for balancing non-preemptive independent tasks. This algorithm gives preference and priority as the main QoS parameter. [4]

IV. PROPOSED WORK

Nowadays as per the market demand the measuring application success as "user interface" is not sufficient. Various application leaders are moving from business-centric metrics to service level management (SLM) [8] so that to bring IT closer to business.
Our goal is to develop a scalable CLOUD solution [6] which is capable of delivering needs of Stock Broking firm with a better performance, scalability and cost effective solution.

A) Features

We are proposing the cloud computing for load balancing with the following enlisted objectives

1) Level Load Balancing on stock application on User side
2) Setup the Cloud computing model and deployment of the related application
3) Measurement of performance evaluation of each node based on various cloud statistics
4) Resource Monitoring of Cloud Nodes using proposed model
5) Deployment of an application war file on cloud nodes considering their CPU,RAM Usage with the help of cloud controller

B) Architecture Of Proposed Work :

Proposed VM load balancing algorithm is used to balance the load in the cloud flow and also used to check the CPU utilization as per the user's request

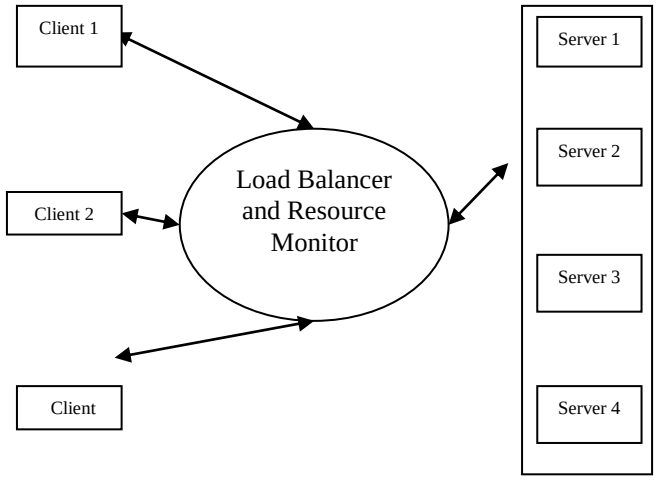


Fig. 2. Proposed Architecture of Load Balancing

C) Details Of Proposed Algorithm

The flow of proposed algorithm is given below

1) Receive request from client
2) Compute execution time of each request on each node n1, n2....
3) Calculation of resource usage threshold for each incoming request
4) If it crosses threshold limit use the resource on another node.
5) Transfer the request to the node whose resource utilization is below threshold value and with less execution time.

We have proposed & implemented the Dynamic load algorithm for effective utilization of CPU & RAM. These are explained in details as below.

D) Dynamic Load Algorithm (DLA)

The DLA algorithm is modified version of Central Scheduler Load Balancing (CSLB) algorithm [4]. The algorithm uses the six phases for load balancing as under.

1) Get Load Status of All the Nodes: Here, we set a scheduler which contains a Monitor to gain and read load status, and also a Database to store the load status and work request historical data of user access to the server. Most of the recent methods of nodes load status collection divided the system resource into several types: CPU utilization, Memory, Disk I/O and network bandwidth etc. But with various size of servers with different services we cannot propose a unified set of those parameters.

2) Evaluate the Status Of nodes: We set a threshold that when the resource utilization beyond the threshold, we can consider compute as a over-load node, also if the resource utilization is under the threshold we know that the node is in a light-load status use and to represent those two statuses.

3) Prediction of Future Load Flow: Based on the statistics, system's load status could show seasonal changes, which help to predict future load of nodes.

4) Estimation of Benefits: When a load status of N is signed as which caused by transient spike, in this condition we cannot make the decision that whether we should perform migration.

5) Select Receiver Nodes: We use the forward probability method to help us to choose a receiver host, every candidate nodes' probability to receive a job or VM is mainly depends on the result of load status evaluation.

6) Relocation: Helps relocating of the heavily loaded nodes to the lighter ones.

V. MATHEMATICAL MODEL

The mathematical model for the entire system which includes selection of Load balancer by the central controller system and then selection of Nodes by the balancer for processing by dynamic load algorithm.

1) Selection of Load Balancer

The function is used for selection of load balancer

$$F = m\_utli(B) = \sum_{n=0}^{n} Memory(n) \qquad (1)$$

Where Memory (n) is memory utilized by node, mem(m)_utli(B) is the memory utilized by Load Balancer.

a. If mem(m)_utli(B) =0 then the Balancer is in Idle state.

b. If Benefit _utli(B) >0 and <= Threshold Load, Balancer is in Normal state.

c. If mem(m)_utli(B) >threshold, Balancer is in Overloaded state.

2) Selection of node in idle state

The selection of node is done as shown in equation (2)

For all node(i) : empty(node(i)) → select(node (i)) (2)

Where i takes the value from a to d for balancer A and a to d for balancer B ,Where a to d are the nodes attached with the balancer.

3) Selection of node in Normal state

Let the system be defined by S for choice of node using Dynamic Load Algorithm on Cloud Partition. The five parameters for the system S selection is given as Input, Output, Function, Success, Failure, hence it is defined as

S = {I, O, F, S, Fi}

Where I= input to the system,
O= output of the system,
F = function used in the system,
S= condition for success,
Fi= condition for failure of the system.

For the given system I = set of jobs that received from the central controller, the output obtained for the system will be (O) = selection of the node, the system is in success state if the node is obtained and the system is in failure state if node is not obtained for storage of job. Selection of Node using Dynamic Load Algorithm the function is computed as

a. Initially node selection is done using randomized selection if memory space is available
b. If the randomly selected node does not have enough space then the node having maximum entry in the forward Pheromone table is selected as shown in equation (3) fp is the table having forward pheromone or value for selection of the next node and max is a variable used to check for maximum pheromone and it is initialized to 0.

$$\sum_{i=1}^{n} fp(i) > \max \begin{cases} Max = fp(i) \\ Next\ node = i \end{cases}$$

VI. PERFORMANCE MEASUREMENT MATRICES

Metrics are a specific calculated measurement. They help us gauge success and failure, as well as enable comparisons. Load testing tools measure key performance statistics and present those metrics to you in graphs and reports.
Focus on the key metrics because reducing data clutter can make it easier to spot bottlenecks and inefficiencies in a web application's design and implementation.

A) Commonly used performance metrics for load testing:

a) Average Response Time

The average time it takes for a server to respond to requests. Usually it is measured in ms. It shows overall performance health from the user perspective.

b) Peak Response Time

The longest response out of all responses for a given time period. It shows outliers, the slowest transactions, the problem areas to investigate.

c) Error Rate

Percentage of failures. Shows what ratio of requests is getting good versus bad responses.

d) Throughput

The data transferred between the user and site. It usually measured in kilobytes per second.

e) Requests per Second

The number of simultaneous hits on the site at approximately the same time.

f) Concurrent Users

The number of users holding a site session, but not necessarily sending requests simultaneously.

g) CPU/Memory Utilization

The percentage of CPU or memory that a web server is using at a point in time. Shows when the system is resource constrained as a root cause of performance failure.

B) Experimental Results

The load test finished with scheduled to run for 20 minutes with a linear pattern, starting at 5 vusers and increasing to 10 vusers.

TABLE I. SUMMARY OF THE RESULT USING LOAD STORM

	Requests	Response (average s)	Response (max s)	RPS (average)	Throughput (average)	Total Transfer
HTML	288	0.6	1.01	0.24	23.05 kB/s	0.3 GB
Other *	1531	0.22	0.72	1.28	10.31 kB/s	0.1 GB
Total	1819	0.28	1.01	1.52	33.36 kB/s	0.4 GB

[a.] *Other includes javascript, css, images, pdf, task migration etc. (any content type except html and xml)

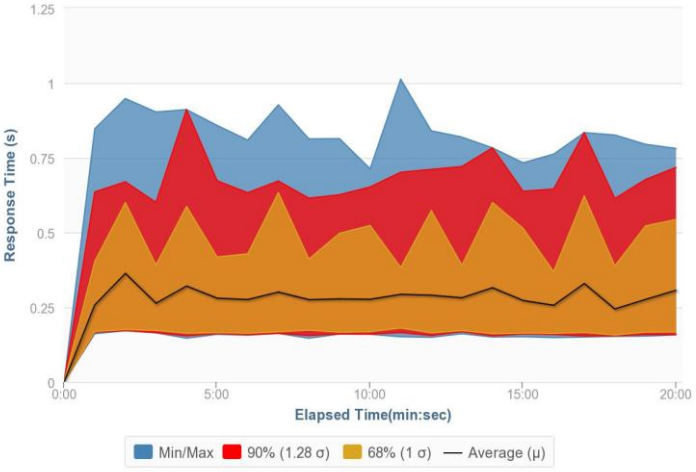


Fig. 3. Overall response time

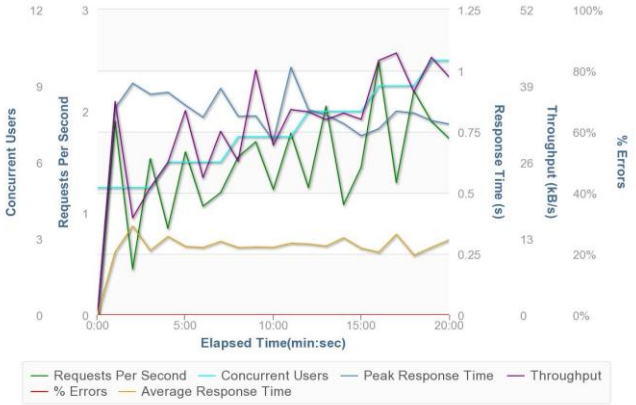


Fig. 4. Summary of load testing

VII. RESULT

A) Overall Response Time Summary

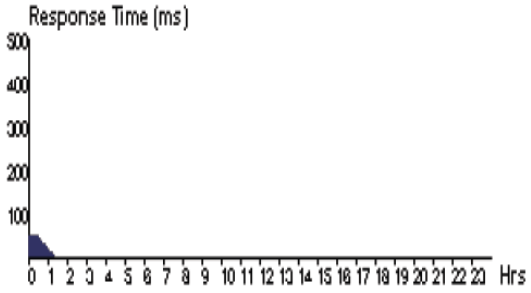
	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	292.82	38.51	609.39
Data Center processing time:	1.02	0.02	2.63

TABLE II. RESPONSE TIME BY REGION

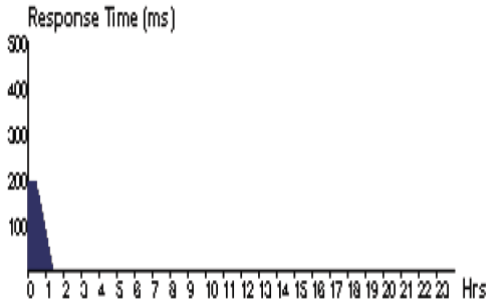
Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	51.00	38.51	62.26
UB2	200.21	154.14	239.33
UB3	299.67	232.84	366.15
UB4	502.27	390.14	609.39
UB5	500.81	375.34	603.77
UB6	200.84	156.89	242.34

B) User Base Hourly Response Times

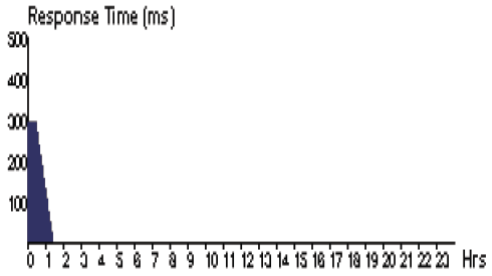
UB1



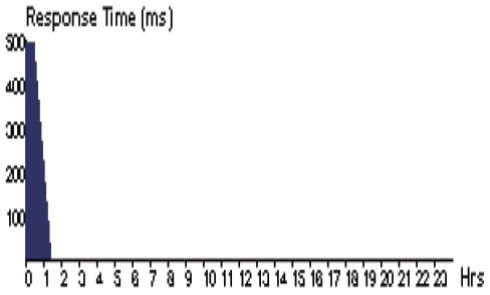
UB2



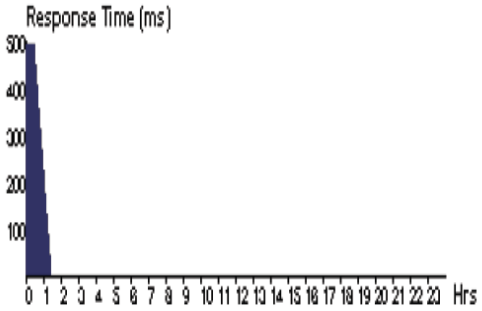
UB3



UB4



UB5



UB6

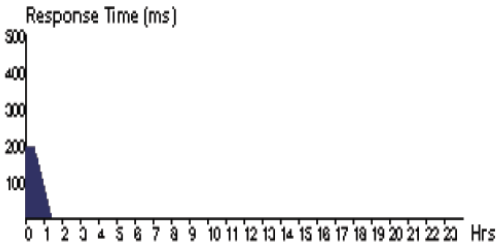


TABLE III. DATA CENTER REQUEST SERVICING TIMES

Data Center	Avg (ms)	Min (ms)	Max (ms)
DC1	0.50	0.04	1.08
DC2	1.36	0.13	2.01
DC3	1.99	0.19	2.63
DC4	0.30	0.02	0.88

C) Data Center Hourly Loading

Cost

Total Virtual Machine Cost ($) : 15.56
Total Data Transfer Cost ($) : 0.3
Grand Total : ($) 15.94

TABLE IV. COST

Data Center	VM Cost $	Data Transfer Cost $	Total $
DC4	0.50	0.10	0.60
DC3	7.53	0.09	7.62
DC2	5.02	0.09	5.11
DC1	2.51	0.10	2.61

VIII. CONCLUSION

Recently Cloud Computing has widely been adopted by the IT industry even though various existing issues like Server Consolidation, Load Balancing, Energy Management, Virtual Machine Migration, etc. The load balancing is essential to distribute the excess dynamic local workload uniformly to all the nodes of Cloud. Due to this its possible to achieve resource utilization ratio. Various available load balancing methods that have been studied & presented by many researchers mainly concentrate on reducing overhead, replication time and overall system performance etc.,. But very few work is done based on the execution time of any task at the run time. Therefore, there is a requirement to develop such load balancing technique that can improve the performance of cloud computing along with maximum utilization resource.

REFERENCES

[1] Chun-Cheng Lin, Hui-Hsin Chin, Der-Jiunn Deng, "Dynamic Multiservice Load Balancing in Cloud-Based Multimedia System", 1932-8184/$31.00 c_ 2013 IEEE, DOI 10.1109/JSYST.2013.2256320.

[2] Yinchuan Deng, Rynson W.H. Lau, "On Delay Adjustment for Dynamic Load Balancing in Distributed Virtual Environments", IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS, VOL. 18, NO. 4, APRIL 2012

[3] Daniel Warneke, Odej Kao, "Exploiting Dynamic Resource Allocation for Efficient Parallel Data Processing in the Cloud", IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 22, NO. 6, JUNE 2011

[3] L.D. Dhinesh Babua, P. Venkata Krishna, "Honey bee behavior inspired load balancing of tasks in cloud computing environments", SciVerse ScienceDirect's Applied Soft Computing, ASOC 1894 1–12, © 2013 Elsevier B.V.

[4] Giuseppe Aceto, Alessio Botta, Walter de Donato, Antonio Pescapè, "Cloud monitoring: A survey", SciVerse ScienceDirect's Computer Networks 57, PP- 2093–2115, ASOC 1894 1–12, © 2013 Elsevier B.V.

[5] Mladen A. Vouk, "Cloud Computing – Issues, Research and Implementations", Proceedings of the ITI 2008 30th Int. Conf. on Information Technology Interfaces, June 23-26, 2008, Cavtat, Croatia

[6] J. Sahoo, S. Mohapatra and R. lath "Virtualization: A survey on concepts, taxonomy and associated security issues" computer and network technology (ICCNT), IEEE, pp. 222-226. April 2010.

[7] G. Pallis, "Cloud Computing: The New Frontier of Internet Computing", IEEE Journal of Internet Computing, Vol. 14, No. 5, September/October 2010, pages 70-73.

[8] A. Khiyati, M. Zbakh, H. El Bakkali, D. El Kettani "Load Balancing Cloud Computing: State Of Art", IEEE, 2012.

[9] Haibo Mi, Huaimin Wang, Hua Cai, Yangfan Zhou3, Michael R Lyu, Zhenbang Chen, "P-Tracer: Path-based Performance Profiling in Cloud Computing Systems", 36th IEE International Conference on Computer Software and Applications, IEEE, 2012.]

[10] Xiaoying Bai, Muyang Li, Bin Chen, Wei-Tek Tsai, Jerry Gao,"Cloud Testing Tools", Proceedings of The 6th IEEE International Symposium on Service Oriented System Engineering, SOSE 2011.

http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6139087

[11] J. Gao, X. Bai, and W. T. Tsai, "Cloud-Testing:Issues, Challenges, Needs and Practice," Software Engineering: An International Journal, vol. 1,no. 1, pp. 9-23, 2011.

[12] PRAKASH.V, BHAVANI.R, "Cloud Testing - Myths and facts and Challenges", International Journal of Reviews in Computing, 10th April 2012. Vol. 9, 2009 - 2011 IJRIC & LLS.

[13] Wei Zhao, Yong Peng, Feng Xie, Zhonghua Dai," Modeling and Simulation of Cloud Computing: A Review", 2012 IEEE Asia Pacific Cloud Computing Congress (APCloudCC), IEEE, 2012.

[14] Brian F. Cooper, Adam Silberstein, Erwin Tam, Raghu Ramakrishnan, Russell Sears, "Benchmarking Cloud Serving Systems with YCSB", in Proceedings of the 1st ACM symposium on Cloud computing, 2010, pp. 143-154.

[15] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms." Software: Practice and Experience, Vol.41, No.1, pp.23-50, 2011.

[16] R. N. Calheiros, R. Ranjan, C. A. F. De Rose, and R. Buyya, "CloudSim: a novel framework for modeling and simulation of cloud computing infrastructure and services," Technical Report, GRIDS-TR-2009-1, Grid Computing and Distributed Systems Laboratory, The University of Melbourne, Australia, 2009.

[17] R. Buyya, R. Ranjan, and R. N. Calheiros, "Modeling and simulation of scalable cloud computing environments and the CloudSim toolkit: challenges and opportunities," The International Conference on Hign Performance Computing and Simulation, pp.1-11, 2009.

[18] Ilango Sriram, "SPECI, a simulation tool exploring cloud-scale data centres", CloudCom 2009, LNCS 5931, pp. 381-392, 2009, M.G. Jaatun, G. Zhao, and C. Rong (Eds.), Springer-Verlag Berlin Heidelberg, 2009

[19] Simon Ostermann, Kassian Plankensteiner, Radu Prodan, and Thomas Fahringer, "GroudSim: An Event-Based Simulation Framework for Computational Grids and Clouds", M.R. Guarracino et al. (Eds.): Euro-Par 2010 Workshops, LNCS 6586, pp. 305-313, 2011. Springer-Verlag Berlin Heidelberg, 2011

[20] B. Wickremasinghe, "CloudAnalyst: a cloudSim-based tool for modeling and analysis of large scale cloud computing environmens, "MEDC Project Report, 2009.