A New Static Load Balancing Algorithm in Cloud Computing

Abhay Kumar Agarwal
Department of Computer Science & Engineering
Kamla Nehru Institute of Technology
Sultanpur

Atul Raj
Department of Computer Science & Engineering
Kamla Nehru Institute of Technology
Sultanpur

ABSTRACT
This paper proposes an algorithm that we named as a New Static load balancing algorithm in cloud computing. The proposed algorithm is using the concept of both Active Monitoring Load Balancing Algorithm and Throttled Load Balancing Algorithm. The detailed design, pseudo code and implementation of algorithm are also presented in this paper. The results (Overall Response Time and Datacenter Processing Time) obtained are compared with the results of Throttled Load Balancing Algorithm. This comparison is done after implementing and analysing each of the existing algorithms discussed in this paper, and found that Throttled Load Balancing Algorithm is best among all the existing. The other sections in the paper are introduction, related works, conclusion etc.

General Terms
Cloud Computing, Load Balancing.

Keywords
datacenter, static load balancing, algorithm.

1. INTRODUCTION
The cloud computing is one of the hot topic now-a- days. Lots of research is going on in this field. There are many issues in the area which are being discussed currently. One such issue is of load balancing in cloud computing. Load Balancing is used for minimizing the total waiting time of the user. In cloud computing load balancing are used for balancing the load on virtual machine and cloud resources. When request generated by users are received by cloud hosting environment, the load balancer (load balancing algorithm) distribute the load over various cloud server so that all server should be utilized efficiently (no server is under loaded or over loaded).

The paper proposes a New Static Load Balancing Algorithm in cloud computing. The idea of proposed algorithm has been taken from two algorithms i.e. Throttled Load Balancing Algorithm and Active Monitoring Load Balancing Algorithm. Proposed algorithm removes the drawback of both. The algorithm is implemented in CloudSim simulator using java. The result obtained shows the proposed algorithm reduces the Overall Response Time and Datacenter Processing Time.

The rest of the paper is organized as follows: related work is presented in section 2. Section 3 presents the comparison of various existing static load balancing algorithm. Section 4 presents the proposed algorithm that includes flowchart and pseudo-code. Experimental setup has been given in section 5 of the paper. Section 6 gives results its comparison and analysis and section 7 concludes the paper.

2. RELATED WORK
This section presents the related works to the load balancing algorithms in clouds. The load balancing in cloud computing is obtained in two ways: Dynamic Load Balancing Algorithm and Static Load Balancing Algorithm. The Dynamic load balancing algorithm does not require prior knowledge of system resources, so that the decision of shifting of the load depends on the current state of system on the other hand the static load balancing algorithm requires prior knowledge of system resources, so that the decision of shifting of the load does not depend on the current state of system. Static algorithm performs better if prior knowledge of server is mentioned. Further from figure 1 we can see classification of both types of algorithms.

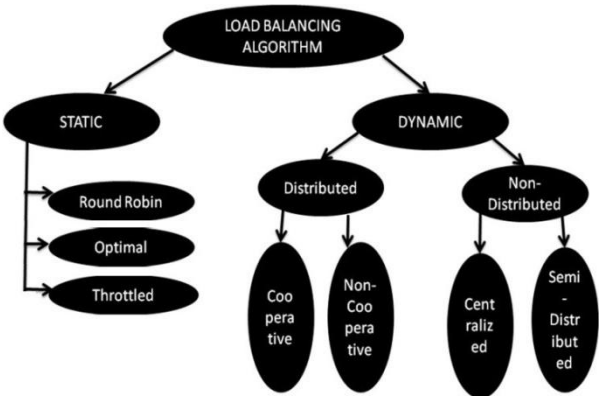

Figure 1: Classification of load balancing algorithms in cloud.

As this paper proposes a new Static load balancing algorithm so we discusses each types of static load balancing algorithms in detail which are:

2.1 Round Robin Scheduling Algorithm
One of the simplest load balancing techniques is Round Robin, in which all processes are divided amid all available processors [2]. The allocation order of processes is maintained locally which is independent of the allocation from the remote processor. In this technique, the request is sent to the node having least number of connections, and because of this at some point of time, some node may be heavily loaded and other remain idle.

Drawback of Round Robin Load Balancing Algorithm

It simply works on time slicing. It allocates the load on various nodes on basis of time without considering the need of resource.

2.2 Active Monitoring Load Balancing (Optimal) Algorithm
In equally spread current execution the random arrival of load in a cloud environment can cause of some the 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 to lightly loaded

servers. Efficient scheduling and resource allocation is a critical characteristic of cloud computing based on which the performance of the system is estimated [4]. The considered characteristics have an impact on cost optimization, which are obtained by improved response time and processing time.

Here the jobs are submitted by the clients to the computing system. As the submitted jobs arrive to the cloud they are queued in the stack. The cloud manager estimates the job size and checks for the availability of the virtual machine and also the capacity of the virtual machine. Once the job size and the available resource (virtual machine) size match, the job scheduler immediately allocates the identified resource to the job in queue [3].

Drawback of Active Monitoring Load Balancing (Optimal) Algorithm

This algorithm checks all virtual machine in sequence for its availability to allocate resource on a free virtual machine.

2.3 Throttled Load Balancing Algorithm

This algorithm is a static load balancing algorithm. Here we first check the index values of all the virtual machine in the system. The request is sent where load balancer parses a table for the allocation of the resources in the system. It assigns the request to a particular load balancer which passes or responds reverse the request to the requester and updates the allocation policy [5]. After the successful allocation of the system the whole process for the de-allocation of the system also starts. This mechanism provides a higher amount of resource sharing and allocation in the system resulting in the higher performance and utilization. The throttling threshold maintained generally is 1. It could be modified easily to make the threshold a configurable value.

This algorithm ensures that pre-defined numbers of cloudlets are allocated to a single VM at any given time. If there are more request groups are present then the number of available VM's at datacenter allocate incoming request. Otherwise queues until the next VM becomes available.

Drawback of Throttled Load Balancing Algorithm

This algorithm is an improvement in Active Monitoring Load Balancing (Optimal) algorithm. This algorithm starts searching from last allocated virtual machine to nth virtual machine. But there is still a problem which is it do not utilises those virtual machine which become free after the execution of request.

3. COMPARISION OF VARIOUS EXISTING STATIC LOAD BALANCING ALGORITHMS

This section presents the comparison between the various existing static load balancing algorithms like, Round Robin Scheduling Algorithm, Optimal Scheduling Algorithm and Throttled Algorithm. The comparison is done on the basis of overall response time and datacenter processing time as shown in table1 and table 2. The values of each for all the algorithms were calculated by implementing them. Each of the algorithms is implemented by taking same hardware, software, simulator and other parameters that is used by us to implement the proposed algorithm.

Table 1: Comparative analysis of overall response time of various load balancing algorithms in cloud computing

Static Time→ Algorithm ↓	Overall Response Time		
	AVERAGE (ms)	***MINIMUM (ms)***	***MAXIMUM (ms)***
Round Robin	351.22	36.86	17755.03
Throttled	240.28	36.36	16395.53
Optimal	352.69	35.63	18052.26

Table 2: Comparative analysis of datacenter processing time of various load balancing algorithms in cloud computing

Static Time→ Algorithm ↓	Datacenter Processing Time		
	AVERAGE (ms)	***MINIMUM (ms)***	***MAXIMUM (ms)***
Round Robin	245.44	0.01	17701.55
Throttled	138.83	0.01	16338.52
Optimal	246.99	0.01	17999

On analyzing experimentally that has been done, it has been found that Throttled Load Balancing Algorithm is best among Round Robin Scheduling Algorithm, Optimal Scheduling Algorithm and Throttled Algorithm.

As we have seen that there exist drawbacks in each algorithm discussed in the previous section. It can also be seen that Throttled Load Balancing algorithm is best among the algorithms discussed. The next section we presents Design of a New Static Load Balancing Algorithm that further improves datacenter processing time and overall time as compare to Throttled Load Balancing Algorithm.

4. DESIGN OF NEW STATIC LOAD BALANCING ALGORITHM

This section presents the flow chart and pseudo code with explanation of proposed algorithm.

The proposed algorithm is using the concept of both Active Monitoring Load Balancing Algorithm and Throttled Load Balancing Algorithm. In this algorithm we create two lists (which is basically hash table) in which we are storing the key of the each virtual machine and associated status (i.e. either as "AVAILABLE" or "BUSY") for each virtual machine (VM). When a request arrives from Userbase then it checks the available list of VM's, if any VM is found available in this list, it allocates the request to that VM and set status of that machine as "BUSY", otherwise, if no VM is free, request waits in queue till any process completes its execution and VM changes its status as "AVAILABLE".

Procedure is repeated until all the requests coming from user base complete their execution. The entire process can also be understood by flow chart given in figure 2.

4.1 Flowchart of New Static Load Balancing Algorithm

Request is generated by a client from a UserBase. A UserBase is a collection of various clients in a particular region. The request generated by a client from a particular UserBase is called a cloudlet. These cloudlets are received by a Datacenter Controller. It keeps all the information about the status of every VM. Datacenter Controller looks for the availability of virtual machines for the cloudlets. If a VM is found as available it is allocated to Cloudlets and Datacenter Controller receives the VMid of that particular VM and sets its status as BUSY. If VM is found as not available then cloudlet waits in waiting queue in Datacenter controller. The VM that completes the processing with cloudlet makes its status again as AVAILABLE.

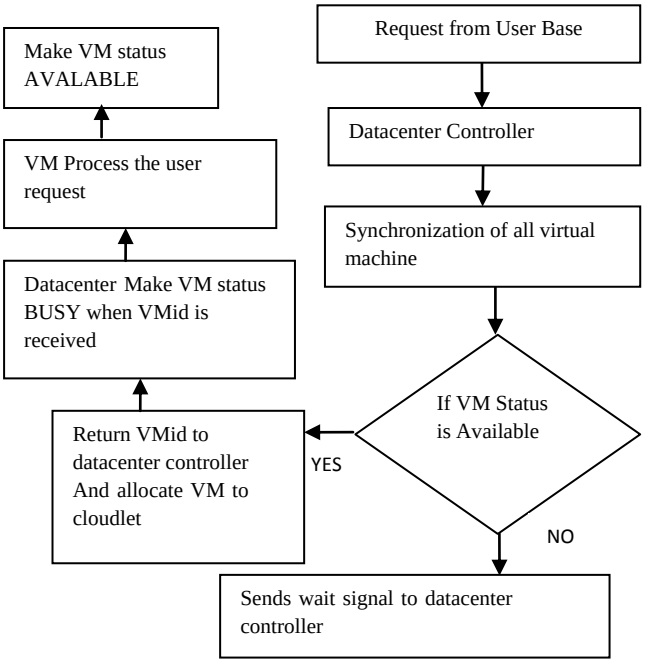

Figure 2: Flow chart of New Static Load Balancing Algorithm

A pseudo code is also presented to well understand the proposed algorithm.

4.2 Pseudo-code of proposed Load Balancing Algorithm

The pseudo code for algorithm1 is for single request (cloudlet). The same algorithm executes for each request arrived at datacenter. The execution of algorithm starts by initializing VM allocation status is 'AVALABLE' in the VM state list as mention in line number 3 of the algorithm. In line number 4, a variable current allocation (count) gets the number of all allocated VMs. Whenever a new request is received by datacenter controller it send the request in queue. Now setting a condition between current allocation count size and VM state list size. If condition found to be true in line number 5.4 then VMstate variable (State) gets the value of flag ('AVALABLE' or 'BUSY') from VM state list. If the value in VMstate variable state is 'AVALABLE' then datacenter controller takes a request from queue to provide VM and the key (VMid) of corresponding allocated VM is passed to datacenter controller. Now datacenter controller makes VM status 'BUSY' of whose VMid is received.

If condition found to be false in the line number 5.4 then request continues to be in wait in the queue in side datacenter controller.

```
1.  New_Algorithm_Load_Balancing
2.  {
3.  Initialize all VM allocation status to AVAILABLE in the
    VM State list;
4.  Current Allocation cout= Synchronized. All allocated
    VM.
5.  While(New request are received by the data centre
    Controller)
  5.1  Do
  5.2  {
  5.3  Data Centre Controller queues the request.
  5.4  If (current Allocation cout.size () < VmStatesList.size
       ())
       5.4.1.  if(state.equals(VirtualMachinestate.Available)
               )
       5.4.2.  {
               5.4.2.1.  Data Centre Controller removes a
                         request from beginning of queue.
               5.4.2.2.  VMid=key;
               5.4.2.3.  VMstatelist.put(VMid.VirtualMachine
                         .BUSY)
               5.4.2.4.  Allocated VM process the request
               5.4.2.5.  VMstatelist.put(VMid.VirtualMachine
                         .AVALABLE
               5.4.2.6.  }
       5.4.3.                  }
  5.5          }
6   }
```

Algorithm 1: New Static Load Balancing Algorithm

The next section presents the detail about the experiment set up. It provides the detail of hardware, software, simulator and other parameters used to perform the implementation of the proposed algorithm.

5. EXPERIMENT SET UP

The implementation of proposed algorithm is done using hardware, software, simulator and other parameters. The detailed about which are:

5.1 Hardware

For the implementation, hardware involves one computer system with the following specifications: processor Dual Core of 2.1 GHz, RAM of 2GB, Cache 2MB.

5.2 Software

The existing and proposed algorithm has been implemented on operating system Windows 7. The language used in implementing algorithms is Java (JDK 1.7) on IDE Eclipse Kepler 2014. Microsoft Excel and MatLab are used to draw graph for showing various results.

5.3 Simulator

CloudSim [6] is the most popular simulator tool available for cloud computing environment. It is an event driven simulator built upon the core engine of grid simulator GridSim [7] Java the most powerful object oriented programming language is being used in CloudSim, because of OOP feature, CloudSim modules can be easily extendable with the user's requirement. CloudSim has feature of modeling and creating a huge datacenter, unlimited number of virtual machines, introducing brokering policy and support the important feature of cloud computing pay-as-you-go model. One of its unique features is federated policy, which is rarely available to any other simulators.

Because of extendibility nature of CloudSim its popularity is being increased day by day. Due to the lack of many important features in new cloud simulators, Cloud Analyst [5], Network CloudSim [8], EMUSIM [9], CDOSim [10] are developed integrating new features to the CloudSim Modules. Currently, in HP labs (Palo Alto) and Duke University (U.S.A.) researchers are using CloudSim for evaluation of resource algorithms and energy-efficient management of datacenters.

Later on many works have been done for the improvement of CloudSim. Yuxiang Shi proposed an energy scheme using "Linear Predicting Method" (LPM) and "Flat Period Reservation-Reduced Method" (FPRRM) based on CloudSim that will reduce the energy consumption of cloud [11]. G. Belalem had made an approach to improve resource allocation scheme in CloudSim [12]. Y. Shi added file stripping and functions for data replication management in CloudSim, Thus a new simulation framework is proposed for data storage processing and computation [13]. One of the drawbacks of CloudSim is lack Of GUI.

6. OTHER PARAMETERS

Number of User Base: 31

Number of Datacenter: 5

Simulation Duration: 60 minutes

Number of Region taken: Globally divided in 6 region

Operating System: LINUX

Virtual Machine Manager: Xen (Para virtualization)

The next section presents the comparison and analysis of the results that are obtained on implementing the three static algorithms.

7. COMPARISON AND ANALYSIS

The comparison of the three static algorithms has been done on the basis of response time of cloudlet and Datacenter processing time. The comparison has been done by calculating minimum time, maximum time and average time for the both cases. Throttled algorithm is found to be best among three. The proposed algorithm is therefore compared throttled on the basis of same parameter as previous. The detail about the results obtained for proposed algorithm and it is comparisons with Throttled algorithm are as follows.

7.1 Overall Response Time of Cloudlet

On seeing the results shown in table 3 it can be analyzed that the time (minimum, maximum and average) taken by proposed algorithm is significantly less as compared to Throttled Load Balancing Algorithm in cloud computing.

The table 3 shows the average value, minimum value and maximum value of overall response time (ms) for the two algorithms. Separate graphs are plotted using the data given in table 3, for each of the average, minimum and maximum overall response time has also been plotted in figure 3, figure 4 and figure 5 respectively.

Table 3: Shows the Overall response time of load balancing algorithms in cloud computing.

Static Time→ Algorithm↓	Overall Response Time		
	AVERAGE (ms)	***MINIMUM (ms)***	***MAXIMUM (ms)***
Throttled	240.28	36.36	16395.53
NSLBA*	238.86	35.12	16395.04

***New Static Load Balancing Algorithm**

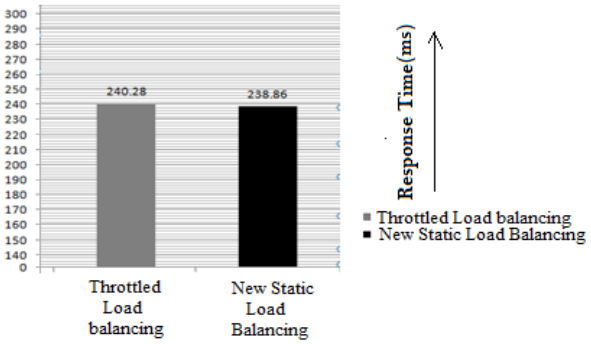

Figure 3: Shows the average case comparison of Overall response time.

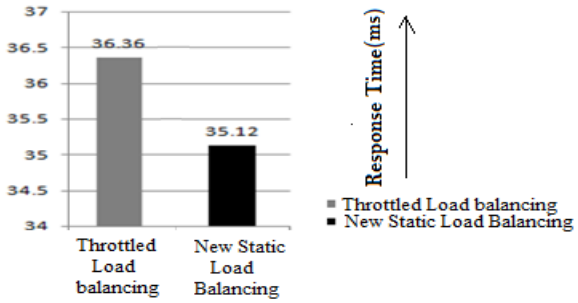

Figure 4: Shows the minimum case comparison of Overall response time.

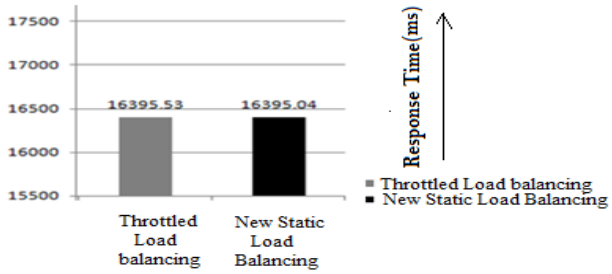

Figure 5: Shows the maximum case comparison of Overall response time.

7.2 Datacenter Processing Time of Cloudlet

On seeing the results shown in table 4 it can be analyzed that the time (minimum, maximum and average) taken by proposed algorithm is significantly less as compared to Throttled Load Balancing Algorithm in cloud computing.

The table 4 shows the average value, minimum value and

maximum value of overall response time (ms) for the two algorithms. Separate graphs are plotted using the data given in table 4, for each of the average, minimum and maximum Datacenter Processing time has also been plotted in figure 6, figure 7 and figure 8 respectively.

Table 4: Shows the Datacenter Processing time of load balancing algorithms in cloud computing.

Static Time→ Algorithm ↓	Datacenter Processing Time		
	AVERAGE(ms)	*MINIMUM (ms)*	*MAXIMUM (ms)*
Throttled	138.83	0.01	16338.52
NSLBA*	132.93	0.01	16335.04

***New Static Load Balancing Algorithm.**

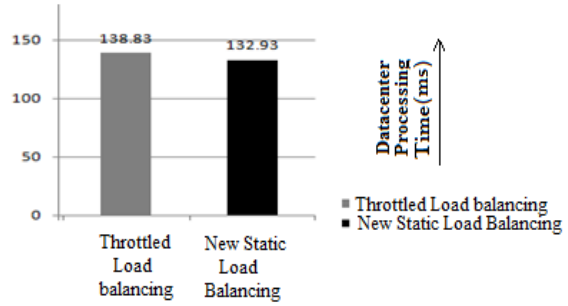

Figure 6: Shows the average case comparison of Datacenter Processing time.

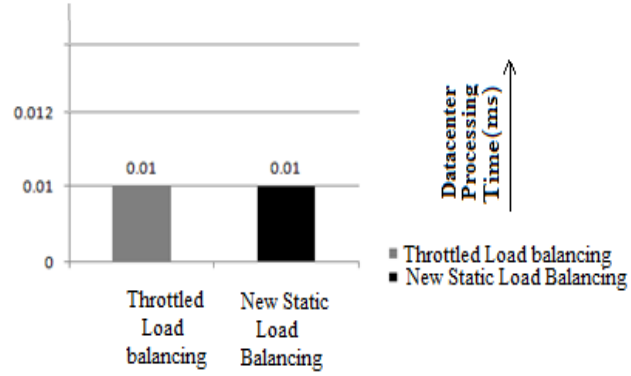

Figure 7: Shows the minimum case comparison of Datacenter Processing time.

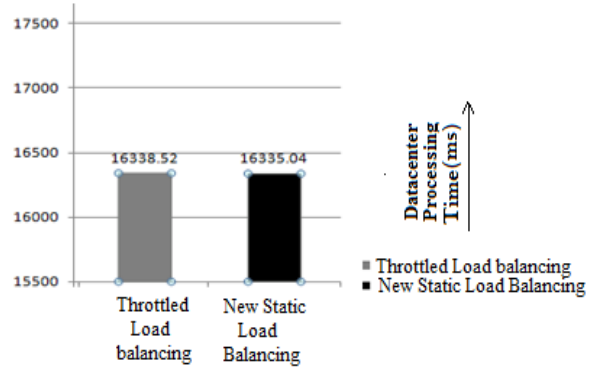

Figure 8: Shows the maximum case comparison of Datacenter Processing time.

Thus it can be seen from the results that the proposed algorithm i.e. new static load balancing algorithm works better than all the other existing static load balancing algorithms in cloud computing for overall response time and datacenter processing time.

8. CONCLUSION

It may be seen from the paper that proposed New Static Load Balancing Algorithm works better than Throttled load balancing algorithm which itself is best among other existing static load balancing algorithms discussed in the paper. It may be seen that overall response time and datacenter processing time (for all cases i.e. average, minimum and maximum) in case of proposed algorithm comes out to be better than Throttled load balancing algorithm.

Since in both the cases that are overall response time and datacenter processing time proposed New Static Load Balancing Algorithm perform better so we can say that our proposed algorithm is best among the other algorithms discussed in this paper.

9. REFERENCES

[1] Soumya Ray and Ajanta De Sarkar, "Execution analysis of load balancing algorithms in cloud computing environment," International Journal on Cloud Computing: Services and Architecture (IJCCSA),Vol.2, No.5, October 2012.

[2] Wei-Tek Tsai, Xin Sun, Janaka Balasooriya "Service-Oriented Cloud Computing Architecture" Computer society 2010.

[3] Pooja Samal1 and Pranati Mishra2, "Analysis of Variants in Round Robin Algorithms for Load Balancing in Cloud Computing", (IJCSIT) International Journals of Computer Science and Information Technologies, Volume 4 (3), 2013, pg. number 416- 419.

[4] Zenon Chaczko, Venkatesh Mahadevan, Shahrzad Aslanzadeh and Christopher Mcdermid," Availability and Load Balancing in Cloud Computing" IPCSIT vol.14 (2011).

[5] Martin Randles, David Lamb, A. Taleb-Bendiab, A Comparative Study into Distributed Load Balancing Algorithms for Cloud Computing, 2010 IEEE 24th International Con-ference on Advanced Information Networking and Applications Workshops

[6] R. N. Calheiros, R Ranjan, A Beloglazov1, C A. F. De Rose, R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms", Published online 24 August 2010 in Wiley Online Library.

[7] R. Buyya,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 2002.

[8] Garg, S. K., & Buyya, R. (2011, December). "NetworkCloudSim: modelling parallel applications in cloud simulations." In Utility and Cloud Computing (UCC), 2011 Fourth IEEE International Conference on (pp. 105-113).

[9] R. N. Calheiros, M .A. S. Netto, C. A. F. De Rose, and R. Buyya,"EMUSIM: an integrated emulation and simulation environment for modeling, evaluation, and validation of performance of cloud computing applications," Software-Practice and Experience, 2012.

[10] F. Fittkau, S. Frey, W. Hasselbring, "CDOSim: Simulating Cloud Deployment Options for Software Migration Support", IEEE 6th International Workshop on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA), 2012.

[11] Y. Shi, X. Jiang, K.Ye, "An Energy-Efficient Scheme for Cloud Resource Provisioning Based on CloudSim", IEEE International Conference on Cluster Computing, 2011.

[12] G. Belalem, F. Z. Tayeb, and W. Zaoui, "Approaches to improve the resource management in the simulator CloudSim," The International Conference on Information Computing and Applications, 2010.

[13] S. Long, Y. Zhao, "A toolkit for modeling and simulating cloud data storage: an extension to CloudSim", International Conference on Control Engineering and Communication Technology (ICCECT), 2012.