Load Balancing in Heterogeneous Cloud Environment

[1] B.Saritha, [2] J.Himabindu Priyanka, [3]N.Krishna Vardhan, [4] K.Sreekanth

[1,2,3,4] Associate Professor, CSE Department, St.Martins Engineering College,JNTUH,India.

Abstract— Cloud computing is a heterogeneous environment offers a rapidly and on-demand wide range of services to the end users.It's a new solution and strategy for high performance computing where, it achieve high availability, flexibility, cost reduced and on demand scalability. The need to efficient and powerful load balancing algorithms is one of the most important issues in cloud computing to improve the performance. This paper proposed a hybrid load balancing algorithm to improve the performance and efficiency in heterogeneous cloud environment. The algorithm considers the current resource information and the CPU capacity factor and takes advantages of both random and greedy algorithms. The hybrid algorithm has been evaluated and compared with other algorithms using cloud Analyst simulator. The experiment results show that the proposed algorithm improves the average response time and average processing time compared with other algorithms.

Keywords—Cloud Computing, Cloud Analyst, Scheduling algorithm, Virtual Machine Load Balancing.

I. INTRODUCTION

In recent years, Cloud computing become a new computing model emerged from the rapidly development of internet. It leads the new IT revolution. Cloud computing considered an evolution of distributed systems. The National Institute of Standards and Technology's (NIST) define a Cloud computing as "cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. "[3] cloud computing has moved computing and data away from desktop and portable PCs into large data centers[2]. Cloud computing is a heterogeneous environment offers a rapidly and on-demand wide range of services[1]. Heterogeneous environment means having different hardware characteristics including CPU, memory, storage and other hardware[2].Load balancing considered as one of the most challenges in cloud computing. It is the major factor to improve the performance of the cloud computing. The current load balance scheduling algorithms in cloud computing environment is not highly efficient[5]. Load balancing in cloud computing environment is very complex task till today, because prediction of user request arrivals on the server is not possible, and each virtual machine has different specification, so it becomes a very difficult to schedule job and balance the load among node[6].

Recently, Many research works have proposed a load balancing algorithms in cloud computing such as Round Robin, Equally Spread Current Execution and Throttled Load Balancing Algorithm. The current load balance scheduling algorithms in heterogeneous cloud computing environment is not highly efficient[5]. This research proposes a hybrid load balancing algorithm to improve the performance and efficiency in heterogeneous cloud computing environment. The proposed algorithm takes advantages of both random and greedy algorithms and considers the current resource information and the CPU capacity factor to achieve the objectives. The hybrid algorithm has been evaluated and compared with other algorithms using cloud Analyst simulator. The result showed improvements on average response time and on processing time by considering the current resource information and the CPU capacity factor compared with other algorithms, and this means the performance has improved. The paper in general will be organized as follows: Section two is devoted to cloud computing overview. In section three the load balancing overview. In section four the related works. In section five we define the proposed algorithm. Section six is about experiment and results. Section seven is for the conclusion and future work.

II. CLOUD COMPUTING

Cloud computing is a heterogeneous environment offers a rapidly and on-demand wide range of services [1]. Heterogeneous environment means having different hardware characteristics including CPU, memory, storage and other hardware [2]. The business owner can start and expand without invest in the infrastructure with lowering operating and maintenance cost. It has moved computing and data away from desktop and portable PCs. The cloud is a virtualization of resources that maintains and manages itself [13]. It builds on a wide range of different computing technologies such as high-performance computing, distributed systems, virtualization, storage, networking, security, management and automation, Service-Oriented Architecture (SOA), Service-Level Agreement (SLA) and Quality of Service (QoS)...etc.[14].

A. Cloud Service Model

Cloud Computing as shown in Figure 1 can be delivered through such delivery models as follow:

- **Infrastructure as a Service (IaaS)**: This model of Cloud computing provide Hardware as a Service via Internet such as storage, CPU and other. There are many IaaS providers such as Amazon Elastic Cloud Compute (EC2), Rackspace[4][5].
- **Platform as a Service (PaaS)**:

Cloud computing provide a platform as a services that required for building application, where user using tools and libraries for Cloud service providers, and also consumers deployed their applications without costing of hardware.

There are many PaaS providers such as Google App Engine, Windows Azure[4][5].

- **Software as a Service (SaaS)**: Focus on providing different software hosted on the Cloud and usually referred to as on-demand software; the consumer will have to pay for usage of software. Usually consumer access to the software via the Internet **[4][5]**. There are many SaaS provider such as Google Apps, SalesForce.com.

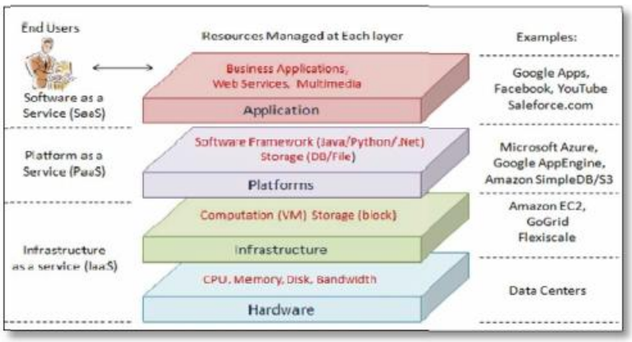

End Users
Resources Managed at Each layer
Examples:
Software as a Service (SaaS)
Business Applications, Web Services, Multimedia
Application
Google Apps, Facebook, YouTube Saleforce.com
Platform as a Service (PaaS)
Software Framework (Java/Python/.Net) Storage (DB/File)
Platforms
Microsoft Azure, Google AppEngine, Amazon SimpleDB/S3
Computation (VM) Storage (block)
Infrastructure
Amazon EC2, GoGrid Flexiscale
Infrastructure as a service (IaaS)
CPU, Memory, Disk, Bandwidth
Hardware
Data Centers

Figure 1 Cloud Computing Architecture

B. Cloud computing model

There are different types of clouds [6], each with its own benefits and drawbacks

- **Public clouds**: A cloud in which service providers offer their resources as services to the general public. Public clouds offer several key benefits to service providers, including no initial capital investment on infrastructure and shifting of risks to infrastructure providers. However, public clouds lack fine-grained control over data, network and security settings, which hampers their effectiveness in many business scenarios[3][7].
- **Private clouds**: Also known as internal clouds, private clouds are designed for exclusive use by a single organization. A private cloud may be built and managed by the organization or by external providers. A private cloud offers the highest degree of control over performance, reliability and security. However, they are often criticized for being similar to traditional proprietary server farms and do not provide benefits such as no up-front capital costs[3][7].
- **Hybrid clouds**: A hybrid cloud is a combination of public and private cloud models that tries to address the limitations of each approach. In a hybrid cloud, part of the service infrastructure runs in private clouds while the remaining part runs in public clouds[3][7].

III. LOAD BALANCING

Load balancing is a process of reassigning the total load to the individual nodes of the collective system to improve both resource utilization and job response time. It also avoiding a situation where some of the nodes are heavily loaded while other nodes are idle or doing very little work. Load balancing ensures that all node in the system does approximately the equal amount of work at any instant of time[2][8]**.** The objective of load balance is to achieve optimal resource utilization, maximize through put, minimum response time, and avoid overload [9]. The heterogeneous environment considered as a major concern[10][11][12]because the heterogeneous environment consist of heterogeneous resource [11], so the behaves of heterogeneous cloud different and has different attributes and different response times for any process [13]. Load balancing algorithm can be divided into two categories as A) Static and B) Dynamic.[14][8].

A. Static algorithms:

Static algorithms divide the traffic equivalently between servers; the load balancing strategy has been made by load balancing algorithm at compile time [15] By this approach the traffic on the servers will be disdained easily and consequently it will make the situation more imperfectly. A general disadvantage of all static schemes is that the final selection of a host for process allocation is made when the process is created and cannot be changed during process execution to make changes in the system load. Round robin algorithms are a static load balance algorithm because the work load distributions between processors are equal [16].

B. Dynamic algorithms:

In dynamic algorithms decisions on load balancing are based on current state of the system. No prior knowledge is needed for load balancing. So it is better than static approach. Dynamic load balancing can be done in two ways: [17][16]

1) Distributed dynamic load balancing:

In the distributed one, the dynamic load balancing algorithm is executed by all nodes present in the system and the task of load balancing is shared among them. A benefit of this is that even if one or more nodes in the system fail, it will not cause the total load balancing process to halt; it instead would affect the system performance to some extent

2) Non-distributed dynamic load balancing:

The interior hardware infrastructure services interrelated to the Clouds is modeled in the simulator by a Data center element for handling service requests. These requests are application elements sand boxed within VMs, which need to be allocated a share of processing power on Data center’s host components. Data center object manages the data center management activities such as VM creation and destruction and does the routing of user requests received from user via the Internet to the VMs.[14].

IV. RELATED WORKS

Many researchers proposed different algorithms in load balancing and job scheduling in Cloud computing , in this section we reflects a number of researches that worked on enhancement of Load Balancing algorithm.

Sethi et al; in [18] introduced a load balancing algorithm using fuzzy logic with Round Robin (RR) algorithm. The algorithm based on various parameters such as processor speed, and assigned load in VM and etc. The algorithm maintains the information of each VM and numbers of requests currently allocated to VM. When a new request is received, the load balancer searches for the least loaded VM and allocate it, but if there are more than one VM, the selection will be based on processor speed and load in VM using fuzzy logic. The results showed that its performance is better than RR algorithm. The drawback of this approach that authors had focused only on how to decrease the response time and they ignored talkabout processing cost. In addition, the researchers compared their results with only RR algorithm which had been enhanced and improved by many researchers before.

Hu et al; in [19] proposed a new algorithm to enhance job scheduling using genetic algorithm. The algorithm used a historical data and current state of the system. And it makes a mapping relationship between the set of physical machines and the set of VMs. It chooses the least-affective solution by computing ahead influence of the system after the deployment of the needed VM resources. They used some equation to find the best scheduling solution using population.

The results showed an improvement in the utilization of resources. On the other hand, the proposed algorithm has high cost to store and retrieve the historical data of the system nodes, and this may also increase the response time and the processing time in a heterogeneous environment consist of heterogeneous resource.

Sharma et al; in [20] proposed a new algorithm to enhance response time of each VM. The proposed algorithm collects information about all VMs in a list and uses it to allocate appropriate VM which status is available. When a new request is received, the load balancer will look at the table and identify VM whose current allocation count is less than max allocation, and then check its status. The result returned to data center and then the data center will allocate this resource to the request. When the VM finished, it will notify the data center to de-allocate it. The proposed algorithm calculates the expected response time of each VM using four equations. The drawback of this algorithm is that it did not provide a solution for server dead lock which has a bad effect on the system performance.

Mishra and et alin [17] proposed an ant colony algorithm as a solution for load balancing in the cloud. Ants depend on the strength of pheromone to select the optimal path that leads to their destination. In the same way each node in the network has a pheromone. Each row in the pheromone table represents the routing preference for each destination, and each column represents the probability of choosing a neighbor as the next hop. If an ant is at a choice point when there is no pheromone, it makes a random decision. If the pheromone is exist, the node with high probability is selected and then the pheromone table is updated by increasing the probability of this node and decreasing other nodes probabilities. The main drawback of this algorithm is that it does not consider the fault tolerance issues.

Singh, Bedi and Gupta in [21] develop a new heterogeneous load balancing algorithms to distribute the load across number of servers, they create VM's of different data center according of host specification including core processor, processing speed, memory, storage etc. Then allocate weighted count according to the RAM allocated to the VM's in the datacenter. Then used a data structure to maintain weight count and the current allocation count of the VM, they allocate the VM which have available status and have a higher RAM. When allocates a new VM, the algorithm returns the VM id to the Data Center Controller, then updates the allocation count for that VM and adding the new allocation to the busy list. When the VM finishes processing the request the algorithm de-allocate the VM and removed the VM from the busy list. The main drawback of the algorithms was the authors did not present any results and any comparison with other algorithms.

V. Hybrid Load Balance Algorithm

In this research we proposed a hybrid algorithm that takes advantages of both random and greedy algorithms; the algorithm adopts the characteristics of randomization and greedy to make an efficient load balancing and covers their disadvantages. The algorithm considers the current resource information and the CPU capacity factor to achieve the objectives. Figure 2 shows the abstract view of proposed algorithm. The hybrid algorithm consists of two main steps:

- In the first step VMs is distribute over hosts according to the host qualifications. The largest number of VMs is located at the most qualified host depending on the Hosts' CPU capacity. For example if we have five VMs and three hosts, where the first host has 1 CPU and its speed = 10000, the second host has 2 CPUs and the speed of every CPU = 10000, and the third host has 3 CPUs and the speed of every CPU = 100000. So, the capacity of the first host = 1*10000=10000, the second host =2*10000=20000 and the third host =3*10000=30000. So according to hosts' capacities; first host will take 1 VM, the second host will take 2 VMs, and the third host which has the largest capacity will take 3 VMs
- In the second step the algorithm used a new index table to record the current loads for each VM. And which used to check the current loads for VM at each iteration, the algorithm read the value of VM load from the index table; when the data center receives a request from the users, it sends the request to the hybrid load balancer. The hybrid algorithm will select k nodes (VM) randomly, and then it will choose the current load for each selectedVM. Then it will choose a VM that have least VM current loads and return the VM id to Data center. The Data center will assign the load to the selected VM and update the value of selected VM in the index table of current loads. Finally when the VM finishes processing the request, it will inform the data center to updating its current load value.

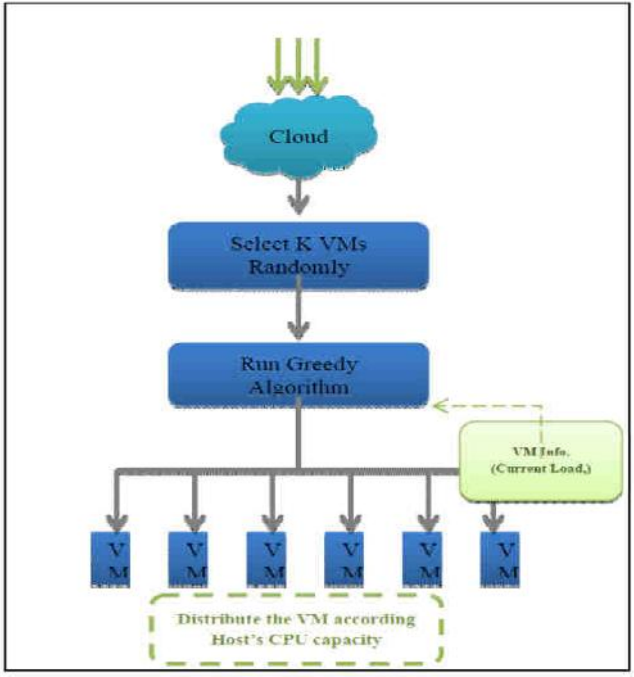


Figure 2 the Proposed Algorithm

A. Pseudo code

The Hybrid algorithm is a load balancing algorithm used by the data center to distribute the received tasks efficiently over the virtual machine under a normal work load by finding the best VM among the group of VMs to assign the load in heterogeneous cloud computing environment. The hybrid algorithm consists of both random and greedy algorithms. The hybrid algorithm considers the current resource information and the CPU capacity factor. The hybrid algorithm will select k nodes (VM) randomly, and choose the current load for each VM selected. Then the hybrid algorithm will choose a VM that have least VM current loads and return the VM ID to Data Center. Figure 3 shows the pseudo code of the proposed algorithm.

```
A Hybrid Algorithm
Input: new request
Output: The VM id that selected to assign the load.
Initialize, Cl_Table(0..n-1) ← 0 At start all VM's have
zero allocation., K← m, VM_id ←-1 , VMids()=-1,i← 0,
currCount ← 0, minCount ← Max_Value, TempVMid ← -
1;
Parses VM_List() to LoadBalancer:
For i← 0 to k //Select VM randomly
   TempVMid ← random(VM_List()).
   VM_id ← TempVMid
If vm_id Exist in Cl_Table(VM_id) then
currCount ← Cl_Table(VM_id)
Else
currCount ← 0
   VMids() ← (VM_id, currCount).
End for
TempVMid ← -1
currCount ← 0
For i ← 0 to k
TempVMid ← i
currCount ← VMids(TempVMid)
If currCount <minCount then
minCount= currCount
    VM_id ← TempVMid
  End if
End for
Cl_Table(VM_id) ← Cl_Table(VM_id)+ 1
```

Figure 3 the hybrid algorithm Pseudo code

VI. Experiment and Results

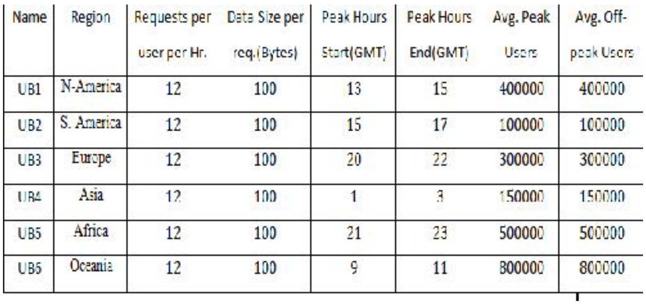
This section presents the experiments and results. Cloud Analyst simulator has been used to compare the proposed hybrid algorithm with the current load balance algorithms. Cloud Analyst is a graphical simulation tool based on Cloudsim for modeling and analysis behavior of cloud computing environment, which supports visual modeling and simulation of large-scale applications that are deployed on Cloud infrastructures[23]. We defined the simulator parameters such as (users configuration, Data centers configuration, VMs configuration). We implemented the hybrid algorithm and the following current load balance algorithms (Round Robin, Equally spread current Execution (ESCE), Random and Greedy algorithms).

A. Experiment

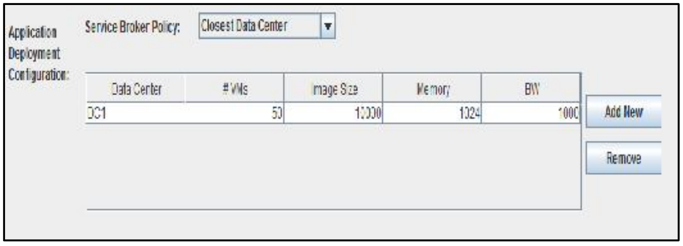
In order to evaluate the proposed hybrid algorithm which consider the capacity of CPU. We run the experiments in heterogeneous environment of hosts; where each machine has different number of CPUs and speed.

In the experiments we set the number of virtual machines in the data center to be 50 VMs and the size used to host applications is 100 MB. Each Virtual machine has 1 GB of RAM memory and 10 MB of available Band width. Simulated hosts is x86 architecture, virtual machine monitor Xen and Linux operating system. The Users are grouped by a factor of 1000, and requests are grouped by a factor of 100. Each user request requires 250 instructions to be executed. The configurations file as in figure 4 and 5, and in Table 1 and 2.

Table1 the user's base configuration



Name	Region	Requests per user per Hr.	Data Size per req.(Bytes)	Peak Hours Start(GMT)	Peak Hours End(GMT)	Avg. Peak Users	Avg. Off-peak Users
UB1	N-America	12	100	13	15	400000	400000
UB2	S. America	12	100	15	17	100000	100000
UB3	Europe	12	100	20	22	300000	300000
UB4	Asia	12	100	1	3	150000	150000
UB5	Africa	12	100	21	23	500000	500000
UB5	Oceania	12	100	9	11	800000	800000

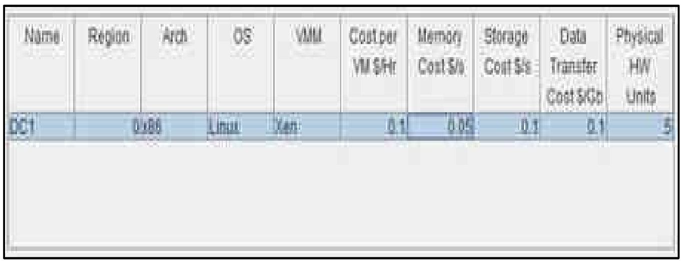


Application Deployment Configuration:

Service Broker Policy: Closest Data Center

Data Center	# VMs	Image Size	Memory	BW
DC1	50	10000	1024	1000

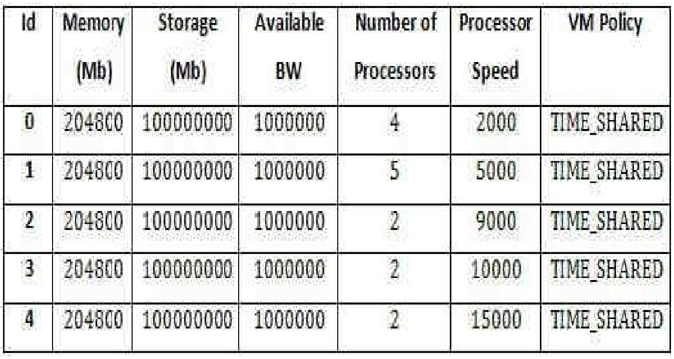
Figure 4 Application deployment configurations



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	5

Figure 5 Data center configuration

Table 2 Hosts Configuration



Id	Memory (Mb)	Storage (Mb)	Available BW	Number of Processors	Processor Speed	VM Policy
0	204800	100000000	1000000	4	2000	TIME_SHARED
1	204800	100000000	1000000	5	5000	TIME_SHARED
2	204800	100000000	1000000	2	9000	TIME_SHARED
3	204800	100000000	1000000	2	10000	TIME_SHARED
4	204800	100000000	1000000	2	15000	TIME_SHARED

B. Results

From this experiment we obtain results as in figure 6:

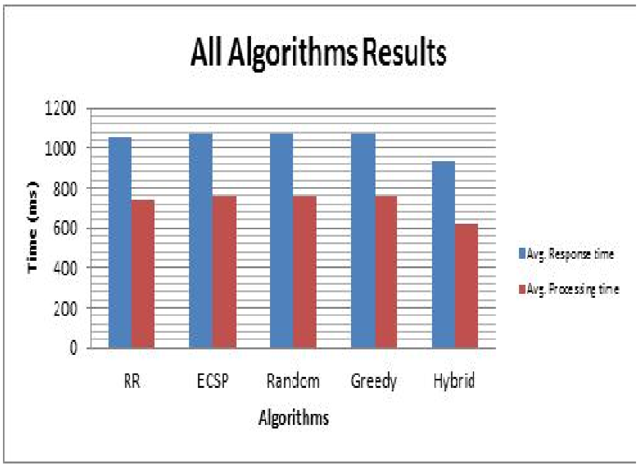


Figure 6 all algorithms results comparison

C. Discussion

In the experiment we have evaluated the proposed hybrid algorithm and compared it with other load balancing algorithms under heterogeneous environment of hosts. The result found that the hybrid algorithm average response time was 930.77(ms) and the average processing time was 620.07(ms) when [K] = 20. This result was better than other algorithms**.** The difference between the results exceeded 100 (ms) on each average response and average processing time. The response time improved because most of selected VM was in the qualified host, and this means the hybrid algorithm add a significant improvement on average response time and on processing time compared with other algorithms. In addition, the performance has improved in heterogeneous cloud computing environment.

VII. CONCLUSION AND FUTURE WORK

Cloud computing is a heterogeneous environment offers a rapidly and on-demand wide range of services to the end users. Load balancing is one of the important issues in cloud computing. The current load balance scheduling algorithms in cloud computing environment is not highly efficient. We proposed a hybrid algorithm to enhance the cloud computing performance. The hybrid algorithm based on randomize and greedy algorithm and considering the current resource information and the CPU capacity factor. The experiments implemented using cloud analyst simulator.

The results showed that the hybrid algorithm add a significant improvements on average response time and average processing time compared with other algorithms. And the performance has improved in heterogeneous cloud computing environment. In future we are going to test this algorithm in a real world for better performance, and we can also consider other parameters for efficient utilization of resources such as consider cost, failover etc. We are going to make change and develop on the algorithm to solve the load balance problem in bursts workload state.

REFERENCES

1. Florence, A.P. and V. Shanthi, Intelligent Dynamic Load Balancing Approach for Computational Cloud. International Journal of Computer Applications, 2013: p. 15-18.
2. Sharma, T. and V.K. Banga, Efficient and Enhanced Algorithm in Cloud Computing. International Journal of Soft Computing and Engineering (IJSCE), March 2013. 3 (1).
3. Zhang, Q., L. Cheng, and R. Boutaba, Cloud computing: state-of-the-art and research challenges. Journal of Internet Services and Applications, 2010. 1(1): p. 7-18.
4. Khatib, V. and E. Khatibi, Issues on Cloud Computing: A Systematic Review, in International Conference on Computational Techniques and Mobile Computing. 2012: Singapore.
5. Sareen, P., Cloud Computing: Types, Architecture, Applications, Concerns, Virtualization and Role of IT Governance in Cloud. International Journal of Advanced Research in Computer Science and Software Engineering, 2013. 3(3): p. 533-538.
6. O., K.S., I. F., and A. O., Cloud Computing Security Issues and Challenges. International Journal of Computer Networks (IJCN), 2011. 3(5): p. 247-255.
7. Sajid, M. and Z. Raza, Cloud Computing: Issues & Challenges, in International Conference on Cloud. 2013. p. 35-41.
8. Mohapatra, S., K.S. Rekha, and S. Mohanty, A Comparison of Four Popular Heuristics for Load Balancing of Virtual Machines in Cloud Computing. International Journal of Computer Applications, 2013. 68.
9. Ray, S. and A. De Sarkar, Execution Analysis Of Load Balancing Algorithms In Cloud Computing Environment. International Journal on Cloud Computing: Services and Architecture (IJCCSA), 2012. 2(5): p. 1-13.
10. Yao, J.H., Ju-hou, Load Balancing Strategy Of Cloud Computing Based On Artificial Bee Algorithm in Computing Technology and Information Management (ICCM). 2012, IEEE: Seoul. p. 185 - 189.
11. Shameem, P.M. and R.S. Shaji, A Methodological Survey on Load Balancing Techniques in Cloud Computing. International Journal of Engineering and Technology (IJET), 2013. 4(5): p. 3801-3812.
12. Behal, V. and A. Kumar, Cloud Computing: Performance Analysis Of Load Balancing Algorithms In Cloud Heterogeneous Environment, in Confluence The Next Generation Information Technology Summit (Confluence). 2014, IEEE: Noida. p. 200 - 205.
13. Kaushik, V.K., H.K. Sharma, and D. Gopalani, Load Balancing In Cloud Computing Using High Level Fragmentation Of Dataset, in International Conference on Cloud, Big Data and Trust. 2013. p. 118-126.
14. Mehta, R., P. Yask, and T. Harshal, Architecture For Distributing Load Dynamically In Cloud Using Server Performance Analysis Under Bursty Workloads. 2012. 1(9).
15. Tiwari, M., K. Gautam, and K. Katare, Analysis of Public Cloud Load Balancing using Partitioning Method and Game Theory. International Journal of Advanced Research in Computer Science and Software Engineering, 2014. 4(2): p. 807-812.
16. Deepika, D. Wadhwa, and N. Kumar, Performance Analysis of Load Balancing Algorithms in Distributed System. Advance in Electronic and Electric Engineering, 2014. 4(1): p. 59-66.
17. Ratan, M. and J. Anant, Ant colony Optimization: A Solution of Load Balancing in Cloud. International Journal of Web & Semantic Technology (IJWesT), 2012. III.
18. Sethi, S., S. Anupama, and K. Jena, S, Efficient load Balancing in Cloud Computing using Fuzzy Logic. IOSR Journal of Engineering (IOSRJEN), 2012. 2(7): p. PP 65-71.
19. Hu, J., et al., A Scheduling Strategy on Load Balancing of Virtual Machine Resources in Cloud Computing Environment, in 3rd International Symposium on Parallel Architectures, Algorithms and Programming. 2010, IEEE. p. 89-96.
20. Sharma, T. and V.K. Banga, Proposed Efficient and Enhanced Algorithm in Cloud Computing. International Journal of Engineering Research & Technology (IJERT), 2013. 2(2).
21. Singh, A., R. Bedi, and S. Gupta, Design and implementation of an Efficient Scheduling algorithm for load balancing in Cloud Computing. International Journal of Emerging Trends & Technology in Computer Science (IJETTCS), 2014. 3(1).
22. cloudsim. cloudbus; Available from: http://www.cloudbus.org/cloudsim/.
23. Pakize, S.R., S.M. Khademi, and A. Gandomi, Comparison Of CloudSim, CloudAnalyst And CloudReports Simulator in Cloud Computing. International Journal of Computer Science And Network Solutions, 2014. 2: p. 19-27.