ISSN No. 0976-5697

Volume 5, No. 4, April 2014 (Special Issue)

International Journal of Advanced Research in Computer Science

REVIEW ARTICLE

Available Online at www.ijarcs.info

Review of Load Balancing Techniques in Cloud Computing Environment: Challenges and Algorithms

G.Narendrababu Reddy
GNITS Hyderabad
gnbreddy25@gmail.com

S.Phani Kumar
GITAM University Hyderabad
phanikumar.s@gitam.edu

Abstract: Cloud computing has become extremely popular by obviating the need for users to own and maintain complex and costly infrastructure. Cloud computing enables customers with limited computational resources to outsource large scale computational tasks to the cloud where massive computational power can be easily utilized in a pay-per-use manner. An internet based development where dynamically scalable and often virtualized resources are provided as a service over the internet has become significant issue. Cloud computing has paved a revolutionary path in this direction of distributed environment for accomplishing optimized performance, quick response time, net work resource utilization, and adaptability of Service Level Agreement (SLA). Cloud computing has multiple benefits as well as it is also accompanied with certain serious technical loopholes. In this paper we focused on one such issue of load balancing. As cloud computing is growing rapidly and clients are demanding more services and better results, load balancing has become a very interesting and important research area. In this paper, we investigate the different algorithms proposed to resolve the issue of load balancing in cloud computing. We discuss and compare various techniques adopted to provide latest approaches in this field.

Keywords: Cloud Computing, Virtualization, Load Balancing, Task scheduling.

I. INTRODUCTION

With the rapid development of processing and storage technologies and evolution of internet, computing resources have become cheaper, more powerful and more ubiquitously available than ever before. This technological trend has enabled the realization of a new computing model called Cloud Computing, in which resources are provided as general utilities that can be leased and released by users through the internet in an on-demand fashion.

According to Rajkumar Buyya et al.[1] "A Cloud is a type of parallel and distributed system consisting of a collection of inter-connected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resource(s) based on service-level agreements established through negotiation between the service provider and consumers." Computing is being transformed to a model consisting of services that are commoditized and delivered in a manner similar to traditional utilities such as water, electricity, gas and telephone. In such a model, users access services based on their requirements without regard to where the services are hosted or how they are delivered.

Cloud computing is an entirely internet based approach where all the applications and files are hosted on a cloud which consists of thousands of computers interlinked together in a complex manner. Cloud computing incorporate concepts of parallel and distributed computing to provide shared resources: hard ware, software and information to computers are other devices on demand. These are provided as a "pay- per- use" model. The emergence of cloud computing [2] has made a tremendous impact on the IT industry over the past few years' where large companies such as Google, Amazon, and Microsoft strive to provide more powerful, reliable and cost–efficient cloud plat forms, and business enterprises seek to reshape their business models to gain benefit from this new paradigm. To better capitalize their investment, the over equipped organizations open their infrastructure to others by exploiting the internet and virtualization technologies using cloud computing model. Virtualization forms the foundation of cloud computing as it provides the capability of pooling computing resources from clusters of servers and dynamically assigning and reassigning virtual resources to applications on-demand.

The rest of this paper is organized as follows. The second section describes the different implementations of load balancing in most used distributed systems, the third section provides challenges of load balancing in cloud computing. In section four we go over the current literature and discuss the algorithms proposed to solve the load balancing issues in cloud computing environment. After that, we discuss and compare the relevant approaches in section five, followed by concluding remarks in section six.

II. OVERVIEW OF LOAD BALANCING

Load balancing [3] 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 and some others are under loaded. The goal of load balancing is improving the performance by balancing the load among the various resources to achieve optimal resource utilization, maximum throughput, shortest response time and avoiding overload. With proper load balancing, resource consumption can be kept to a minimum which will further reduce energy consumption.

To distribute load on different systems we use generally traditional algorithms like those used in web servers, but these algorithms do not always give the expected performance with large scale and distinct structure of service oriented data centers [4]. To overcome the shortcomings of these algorithms, load balancing has been widely studied by

CONFERENCE PAPER
Two day National Conference on Innovation and Advancement in Computing
Organized by: Department of IT, GITAM UNIVERSITY Hyderabad (A.P.) India
Schedule: 28-29 March 2014

researchers and implemented by computer vendors in distributed systems.

In general load balancing algorithms follow two major classifications [5]:

a. Depending on how the charge is distributed and how processes are allocated to nodes(system load);
b. Depending on the information status of the nodes (system topology).

In the first case it designed as central approach, distributed approach or hybrid approach, in the second case as static approach, dynamic or adaptive approach [6].

A. Classification according to the System Load:

a. Centralized approach: In this approach, a single node is responsible for managing the distribution within the whole system.
b. Distributed approach: In this approach, each node independently builds its own load vector by collecting the load information of other nodes. Decisions are made locally using local load vectors. This approach is more suitable for widely distributed systems such as cloud computing.
c. Mixed approach: A combination of above two approaches to take advantage of each approach.

B. Classification according to the System Topology:

a. Static approach: This approach is generally defined in the design or implementation of the system.
b. Dynamic approach: This approach takes into account the current state of the system during load balancing decisions. This approach is more suitable for widely distributed systems such as cloud computing.
c. Adaptive approach: This approach adapts the load distribution to system status changes, by changing their parameters dynamically and even their algorithms. This approach is able to offer better performance when the system state changes frequently [6], [7].

Table I. Load Balancing Metrics

Metric	***Illustration***
Response Time	It is the amount of time taken to respond by a particular load balancing algorithm in a distributed system. This parameter should be minimized.
Throughput	It is used to calculate the number of tasks whose execution has been completed. It should be high to improve the performance of the system.
Resource utilization	It is used to check the utilization of resources. It should be optimized for an efficient load balancing.
Overhead	It determines the amount of overhead involved while implementing a load balancing algorithm. It is composed of overhead due to movement of tasks, inter-processor and inter-process communication. This should be minimized so that a load balancing technique can work efficiently.
Fault Tolerance	It is the time to migrate the jobs or resources from one node to other. It should be minimized in order to enhance the performance of the system.
scalability	It is the ability of an algorithm to perform load balancing for a system with any finite number of nodes. This metric should be improved.
performance	It is used to check the efficiency of the system. This has to be improved at a reasonable cost, e.g., reduce task response time while keeping acceptable delays.

III. LOAD BALANCING CHALLENGES IN CLOUD COMPUTING

Although cloud computing has been widely adopted, research in cloud computing is still in early stages and some scientific challenges remain unsolved by the scientific community, particularly load balancing challenges.

A. Spatial Distribution of Cloud nodes:

Some algorithms are designed to be efficient only for an intranet or closely located nodes where communication delays are negligible. However, it is a challenge to design a load balancing algorithm that can work for spatially distributed nodes. There is a need to develop a way to control load balancing mechanism among all the distributed nodes while being able to effectively tolerate high delays [8].

B. Storage/Replication:

A full replication algorithm does not take efficient storage utilization into account. This is because the same data will be stored in all replication nodes. Full replication algorithms impose higher costs since more storage is needed. However, partial replication algorithms could save parts of the data sets in each node with certain level of overlap based on each node's capabilities such as processing power and capacity [9]. This could lead to better utilization, yet it increases the complexity of load balancing algorithms as they attempt to take into account the availability of the data set's parts across the different cloud nodes.

C. Algorithm Complexity:

Load balancing algorithms are preferred to be less complex in terms of implementation and operations. The higher implementation complexity would lead to a more complex process which could cause some negative performance issues. Furthermore, when algorithms require more information and higher communication for monitoring and control, delays would cause more problems and efficiency will drop. Therefore, load balancing algorithms must be designed in the simplest possible forms [10].

D. Point of Failure:

Controlling the load balancing and collecting data about the different nodes must be designed in a way that avoids having a single point of failure in the algorithm. Some algorithms (centralized algorithms) can provide efficient and effective mechanisms for solving the load balancing in a certain pattern. However, they have the issue of one controller for the whole system. In such cases, if the controller fails, then the whole system would fail. Any load balancing algorithm must be designed in order to overcome this challenge [11]. Distributed load balancing algorithms see to provide a better approach, yet they are much more complex and require more coordination and control to function correctly.

E. Energy Management:

The benefits that advocate the adoption of the cloud is the economy of scale. Energy saving is a key point that allows a global economy where a set of global resources will be supported by reduced providers rather that each one has its own resources. Small data centers can be more beneficial, cheaper and less energy consumer than large datacenter. Small providers can deliver cloud computing services leading to geo-diversity computing. Load balancing will

become a problem on a global scale to ensure an adequate response time with an optimal distribution of resources.

IV. LOAD BALANCING ALGORITHMS REVIEW

In this section we discuss the most known contributions in the literature for load balancing in Cloud computing. We first discuss the static load-balancing algorithms that have been developed for cloud computing. Then, we will discuss the dynamic load-balancing algorithms.

A. Static Load Balancing Algorithms:

The decisions related to balancing of load will be made at compile time when resource requirements are estimated. These algorithms assign the tasks to the nodes based on the ability of the node to process new requests. The process is based solely on prior knowledge of the node's properties and capabilities. The advantage of these algorithms is the simplicity with respect to both implementation and overhead, since there is no need to constantly monitor the nodes for performance statistics. Static algorithms do not consider dynamic changes of the load during run time.

Radojevic et al. [12] suggested an algorithm called CLBDM (Central Load Balancing Decision Model). This is an improvement of the Round Robin algorithm which is based on session switching at the application layer. The improvement done in CLBDM is that the connection time between the client and the node in the cloud is calculated and if the connection time exceeds a threshold then there is an issue. If an issue is found, the connection will be terminated and the task will be forward to another node using the regular RR rules. CLBDM acts as an automated administrator. The idea was inspired by the human administrator point of view.

Nishant Kumar et al. [13] proposed an improvement version of the algorithm presented in [14]. Both algorithms are using ant's behavior to gather information about the cloud nodes to assign the task to a specific node. However, the algorithm in [14] has the ants' synchronization issue and the authors in [13] are trying to solve this issue by adding the feature 'suicide' to the ants. Both algorithms work in the following way, once a request is initiated the ants and pheromone are initiated and the ants start their forward path from the 'head' node. A forward movement means that the ant is moving from one over loaded node looking for the next node to check if it is over loaded or not. Moreover, if the ant finds an under loaded node, it will continue its forward path to check the next node. If the next node is an over loaded node, the ant will use the backward movement to get to the previous node. The improvement of the algorithm proposed in [13] is that the ant will commit suicide once it finds the target node, which will prevent unnecessary backward movements.

The algorithm proposed in [15] is an addition to the Map Reduce algorithm [16]. Map Reduce is a model which has two main tasks: It maps tasks and Reduce tasks results. There are three methods in this model. They are part, comp and group. Map Reduce first executes the part method to initiate the mapping of tasks. At this step the request entity is partitioned into parts using map tasks. Then the key of each part is saved into a hash key table and comp method does the comparison between the parts. After that, the group method groups the parts of similar entities using the reduce tasks. Since several map tasks can read entities in parallel and process them, this will cause the Reduce tasks to be overloaded. Therefore, it is proposed to add one more load balancing level between the map tasks and reduce task to decrease the overload on these tasks. The load balancing in the middle divides only the large tasks into smaller tasks and then the smaller blocks are sent to the reduce tasks based on their availability.

Junjie proposed a load balancing algorithm [17] for the private cloud using virtual machine to physical machine mapping. The architecture of the algorithm contains a central scheduling controller and a resource monitor. The scheduling controller does all the work for calculating which resource is able to take the task and then assigning the task to that specific resource. The resource monitor does the job of collecting the details about the resource availability. The process of mapping tasks goes through for main phases, which are: accepting the virtual machine request, then getting the resources details using the resource monitor. After that, the controller calculates the resources ability to handle tasks and the resource that gets the highest score is the one receiving the task. Finally, the client will be able to access the application.

B. Dynamic Load Balancing Algorithms:

Dynamic load balancing algorithms take into account the different attributes of the node's capabilities and network bandwidth. Most of these algorithms rely on a combination of knowledge based on prior gathered information about the nodes in the cloud and run-time properties collected as the selected nodes process the task's components. These algorithms assign the tasks and may dynamically reassign them to the nodes based on the attributes gathered and calculated. Such algorithms require constant monitoring of the nodes and task progress and are usually harder to implement. However they are more accurate and could result in more efficient load balancing.

In [18] the goal is to find an algorithm to minimize the data duplication and redundancy. The algorithm proposed is called INS (Index Name Server) and it integrates de-duplication and access point selection optimization. There are many parameters involved in the process of calculating the optimum selection point. Some of these parameters are the Hash code of the block of data to be downloaded the position of the server that has the target block of data, the transition quality which is calculated based on the node performance and a weight judgment chart, the maximum bandwidth of downloading from the target server and the path parameter.

Ren [19] presented a dynamic load balancing algorithm for cloud computing based on an existent algorithm called WLC (weighted least connection). The WLC algorithm assigns tasks to the node based on the number of connections that exist for that node. This is done based on a comparison of the SUM of connections of each node in the cloud and then the task is assigned to the node with least number of connections. The proposed algorithm is called ESWLC (Exponential Smooth Forecast based on Weighted Least Connection). ESWLC improves WLC by taking into account the time series and trails. That is ESWLC builds the conclusion of assigning a certain task to a node after having a number of tasks assigned to that node and getting to know the node capabilities. ESWLC builds the decision based on the experience of the node's CPU power, memory, number of connections and the amount of disk space currently being

CONFERENCE PAPER
Two day National Conference on Innovation and Advancement in Computing
Organized by: Department of IT, GITAM UNIVERSITY Hyderabad (A.P.) India
Schedule: 28-29 March 2014

used. ESWLC then predicts which node is to be selected based on exponential smoothing.

Dinesh et al. [20] proposed an algorithm HBBLB (Honey Bee Behavior inspired Load Balancing). Here in this algorithm well load balance across the virtual machines for maximizing the throughput. The load balancing in cloud computing can be achieved by modeling the foraging behavior of honey bees. This algorithm is derived from the behavior of honey bees that uses the method to find and reap food. In bee hives, there is a class of bees called the scout bees and another type was forager bees. The scout bee which forage for food sources, when they find the food, they come back to the beehive to advertise this news by using a dance called waggle/tremble/vibration dance. The purpose of this dance, gives the idea of the quality and/or quantity of food and also its distance from the beehive. Forager bees then follow the Scout Bees to the location that they found food and then begin to reap it. After that they return to the beehive and do a tremble or vibration dance to other bees in the hive giving an idea of how much food is left.

The tasks removed from the overloaded VMs act as Honey Bees. Upon submission to the under load VM, it will update the number of various priority tasks and load of tasks assigned to that VM. This information will be helpful for other tasks, i.e., whenever a high priority has to be submitted to VMs, it should consider the VM that has a minimum number of highpriority tasks so that the particular task will be executed earlier. Since all VMs are sorted in an ascending order, the task removed will be submitted to under loaded VMs. Current workload of all available VMs can be calculated based on the information received from the data center. Advantages are maximizing the throughput; waiting time on task is minimum and overhead become minimum. The disadvantage is if more priority based queues are there then the lower priority load can be stay continuously in the queue.

The paper in [21] proposes an algorithm called Load Balancing Min-Min (LBMM). LBMM has a three level load balancing frame work. It uses the Opportunistic Load Balancing algorithm (OLB) [22]. OLB is a static load balancing algorithm that has the goal of keeping each node in the cloud busy. But OLB does not consider the execution time of the node. LBMM improves OLB by adding a three layered architecture to the algorithm. The first level of the LBMM architecture is the request manager which is responsible for receiving the task and assigning it to one service manager in the second level. When the service manager receives the request, it divides it into subtasks to speed up processing that request. A service manager would also assign the subtask to a service node which is responsible for executing the task. The service manager assigns the tasks to the service node based on different attributes such as the remaining CPU space (node availability), remaining memory and the transmission rate.

V. DISCUSSION AND COMPARISION

In this section we discuss the different algorithms that were discussed in section IV. We also compare these algorithms based on the challenges discussed in section III.

As discussed earlier, the different approaches offer specific solutions for load balancing that suit some situations but not others. The statics algorithms are usually very efficient in terms of overhead as they do not need to monitor the resources during run-time. So they would work very well in a stable environment where operational properties do not change over time and loads are generally uniform and constant. The dynamic algorithms on the other hand offer a much better solution that could adjust the load dynamically at run-time. However this feature leads to high overhead on the system as constant monitoring and control will add more traffic and may cause more delays. Some newly proposed dynamic load balancing algorithms tries to avoid this overhead by utilizing novel task distribution models.

Table II shows a comparison among the review algorithms. The comparison shows the positives and negative points of each algorithm. For example, the INS algorithm is able to handle the load balancing dynamically. However, the provided algorithm is complicated which could cause high implementation complexity. Furthermore, the CLDBM algorithm solves the problem of having a human administrator needed all the time to control the system, by providing a centralized controller. But, if the centralized controller fails any time the whole system will not be able to operate, which will cause a system failure. Having a backup of the central controller could solve the issue for CLDBM in case of failure. As for the ant colony approach, we can see that the decentralized approach provides a good solution to the single point of failure issue. But it could easily cause a network overhead due to large number of dispatched ants. This algorithm can be further improved by introducing better evaluation mechanisms that take into consideration the status of the node and its current available resources. HBBLB mechanism can maximize the throughput by reducing the task waiting time in the queue with minimum overhead. But in this algorithm low priority task continuously stay in the queue. OLB might cause the tasks to be processed in a slower manner and will cause some bottle necks since requests might be pending waiting for nodes to be free. This can be avoided by adding a three layered architecture to the OLB which is LBMM.

Table III illustrates a comparison between the reviewed algorithms in terms of the challenges discussed in section III.

Table II. Merits and demerits of Load Balancing Algorithms

Algorithms	*Merits*	*Demerits*
INS	• Initially proved to handle some sort of dynamic load balancing.	• No forecasting algorithm to identify the future behavior of the nodes. • Complicated in terms of implementation. • Only certain parameters are considered such as distance and time.
ESWLC	• More accurate results than WLC.	• Complicated. • Predication algorithm requires existing data and long processing time.

CLDBM	• Solves issues of Round Robin algorithm. • Automated tasks forwarding reduce the need for a human administrator.	• Inherits Round Robin issues such as not talking into consideration node capabilities. • Single point of failure' if CLBDM fails' the whole process fails. • The threshold might not be applied to all cases.
ANT COLONY	• Best case scenario is that the under loaded node is found at beginning of the search. • Decentralized' no single point of failure. • Ants can collect the information faster.	• Network overhead because of the large number of ants. • Points of initiation of ants and number of ants are not clear. • Nodes status change after ants' visits to them is not taken into account. • Only availability of node is being considered' while there are other factors that should be taken into consideration.
Enhanced Map Reduce	• Less overhead for the reduce tasks.	• High processing time. • Reduce tasks capabilities are not taken into consideration.
VM Mapping	• Reliable calculation method.	• Single point of failure. • Does not take into account network load' and node capabilities.
HBB	• Maximizing the throughput. • Waiting time of the task is minimum. • Low overhead.	• Low priority load become continuously stay in the queue.
LBMM	• Reliable	• Slower than other algorithms because work must pass through three layers to be processed.

Table III. Comparision of Load Balancing Algorithms

	Replication	***Speed***	***Heterogeneity***	***SPOF***	***Network Overhead***	***Spatially Distributed***	***Implementation Complexity***	***Fault tolerance***
INS	Partial	Moderate	Yes	Yes	Yes	Yes	High	No
ESWLS	Full	Fast	Yes	No	Yes	Yes	High	Yes
CLBDM	Full	Slow	Yes	Yes	Yes	Yes	Low	No
Ant colony	Full	Fast	No	No	Yes	No	No	Yes
Map Reduce	Full	Slow	Yes	No	Yes	Yes	High	Yes
VM Mapping	Full	Fast	Yes	Yes	Yes	No	High	Yes
HBB	Full	Fast	Yes	No	Yes	Yes	Low	Yes
LBMM	Full	slow	yes	No	Yes	Yes	Low	No

VI. CONCLUSION

In last few years we assist to emergence of cloud computing model which will rapidly changes the landscape of information technology. However, despite the significant benefits offered by cloud computing, the current technologies are not enough mature. Many key challenges in this domain should be addressed by research community. In this paper, we surveyed multiple load balancing algorithms for cloud computing. We discussed the challenges that must be addressed to provide the most suitable and efficient load balancing algorithms. We also discussed advantages and disadvantages of these algorithms. Then we compared the existing algorithms based on the challenges we discussed.

There is a need to develop an energy efficient load balancing technique that can improve the performance of cloud computing by balancing the work load across all the nodes in the cloud along with maximum resource utilization, in turn reducing energy consumption and carbon emission to an extent which will help to achieve Green Computing.

VII. REFERENCES

[1] Rajkumar Buyya, C.S.Yeo,S.Venugopal,J.Broberg, I.Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility" in : Future Generation Computer Systems vol. 25, 2009, pp.599-616

[2] Qi Zhang, Lu Cheng, Raouf Boutaba, "Cloud computing: state of the art and research challenges" in: J Internet Serv Appl 2010, pp.7-18

[3] K.Ramana, A.Subrayanam and A.Ananda Rao, "Comparative Analysis of Distributed Web Server System Load Balancing Algorithms Using Qualitative Parameters"' VSRD-IJCSIT, vol.1(8),2011, pp.592-600.

[4] Yi Lu, Qiaomin Xie, A.Geller, J R.Larus,A.Greenberg: "A Novel Load Balancing Algorithm for Dynamically Scalable Web Services, IFIP PERFORMANCE 2011, 29th International Symposium on Computer Performance, Modelling, Measurements and Evaluation 2011, 18-20 October 2011, Amsterdam, Netherlands.

[5] Elarbi Badidi, "Architecture of service oriented distribution of objects". Doctoral Thesis 20 July 2000.

[6] N.Shivaratri, P.Krueger and M.Singhal, "Load distributing for locally distributed systems", IEEE Computer 25(2),pp.33-44, December 1992.

[7] T.L Casavant and J.G Kuhl, "A Taxonomy of Scheduling in General Purpose Disrtributed Computing Systems". IEEE Transcations on Software Engineering,14(2),pp.141-154, February 1988.

[8] Buyya R, Ranjan.R and RN Calheiros, "InterCloud: Utility-oriented federation of cloud computing environments for scaling of application services" in proc. 10th International

Conference on Algorithms and Architectures for Parallel Processing, Busan, South Korea 2010.

[9] Foster .I, Y.Zhao, I.Raicu and S.Lu, "Cloud computing and Grid Computing 360-degree compared" in proc. Grid Computing Environments workshop, pp.99-106, 2008.

[10] Grosu.D, A.T. Chronopouls and M.Leung, "Cooperative load balancing in distributed systems" 'in Concurrency and Computation: Practice and Experience' Vol.20 No.16, pp.1953-1976' 2008.

[11] Ranjan.R,L.Zaho,X.Wu,A.Liu; A.Quiroz and M.Parashar, "Peer-to-peer cloud provisioning: Service discovery and load balancing" in Cloud Computing- Principles, Systems and Applications, pp:195-217,2010.

[12] Radojevic.B and M.Zagar, "Analysis of issues with load balancing algorithms in hosted(cloud)environments." In proc. 34th International Convention on MIPRO, IEEE 2011.

[13] Nishant K.P Sharma, V.Krishna, C.Gupta, K.P Singh, N.Nitin and R.Rastogi, "Load Balancing of Nodes in Cloud using Ant Colony Optimization." In proc. 14th International Conference on Computer Modelling and Simulation (UK Sim), IEEE, pp:3-8, March 2012.

[14] Zhang. Z and X.Zang, "A load balancing mechanism based onant colony and complex network theory in open cloud computing federation". In proc. 2nd International Conference on Industrial Mechatronics and Automation (ICIMA), IEEE vol.2, pp:240-243, May 2010.

[15] Kolb.L, A.Thor and E.Rahm, "Load Balancing for MapReduce based Entity Resolutiuon",in proc. 28th "International Conference on Data Engineering (ICDE)", IEEE, pp.618-629,2012.

[16] Gunarathne.T, T-L Wu and G.Fox, "MapReduce in the Clouds for Science" in proc. 2nd International Conference on Cloud Computing Technology and Science (CloudCom),IEEE pp:565-572 November 2010.

[17] Ni.J, Y.Huang, Z.Luan and D.Qian, "Virtual machine mapping policy based on load balancing in private cloud environment" in proc. International conference on Cloud and Service Computing (CSC),IEEE, pp:292-295, December 2011.

[18] T-Y, W-T. Lee, Y-S Lin, H-L Chan and J-S Huang, "Dynamic load balancing mechanism based on cloud storage" in proc. Computing, Commuctions and Applications Conference (ComComAp), IEEE, pp:102-106, January 2012.

[19] Ren.X, R.Lin and H.Zou, "A dynamic load balncing strategy for cloud computing platform based on exponential smoothing forecast" in proc. International Conference on Cloud computing and Intelligent systems (CCIS), IEEE, pp:220-224, September 2011.

[20] Dinesh Babu L.D, P.Venkata Krishna, "Honey Bee Behavior inspired Load Balancing of tasks in Cloud Computing environments",Applied Soft Computing, 13 (2013) pp:2292-2303.

[21] Wang, S-C, Yan, W-P Liao and S-SWang, " Towards a load balancing in a three level cloud computing network" in proc. 3rd International Conference on Computing Science and Information Technology (ICCSIT), IEEE, pp:108-113,July 2010.

[22] Sang.A, X.Wang, M.Madihian and RD Gitlin, "Coordinated load balancing' handoff/cell-site selection and scheduling in multi-cell packet data systems" in wireless Networks, vol.14, No.1, pp:103-120, January 2008.

[23] K Al Nuami, Nader M, M Al Naumi and J Al Jaroodi, "A survey of load balancing in cloud computing", in 2nd Symposium on Network cloud computing and Applications, IEEE, 2012.

[24] A Khiyaita, M.Zbakh, H El Bakkali and Dafir El Kettani "Load Balancing Cloud Computing: State of Art", in IEEE Computer 21(10),pp:106-109,2012.

[25] Vlad nae, Alexandru Iosup, Radu Prodan, Dynamic Resource Provisioning in Massively Multiplayer Online Games,Transcations on Parallel and Distributed Systems ,2010.

[26] Patel.P, and Singh.A.K, "A survey on Resource Allocation Algorithms in Cloud Computing Environment. Golden Research Thoughts, Vol.2 Issue 4, October 2012, ISSN:2231-5063.

[27] Sosinsky.B, "Cloud Computing Bible", Wiley Publishing Inc. ,2012.

[28] P.Mell and T.Grance. NIST Working Defination on Cloud Computing, National Institute of Standars and Technology(NIST). [Onle Document] Availabel at http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf

[29] Begum.S and Prashanth C.S.R, "Review of Load Balancing in Cloud Computing" in International Journal Of Computer Science Issues(IJCSI), vlo.10,Issue 1(2), January 2013, ISSN:1694-0814.

[30] S.Garg and R.Buyya, Green Cloud Computing and Environmental Sustainability, Harnessing Green IT:Principles and Practices, Wiley Press, UK,2011.