A HYBRID CLOUD LOAD BALANCING MODEL USING PARTITIONING METHOD AND SWITCHING MECHANISM

K.ASHOKKUMAR AND R.YAMINI PRIYADHARSHINI
Department of Computer Science, Sathyabama University, Chennai, India
kumar.kashok@gmail.com, ryammy10@gmail.com

ABSTRACT: Cloud computing is a distributed computing network and has the ability to run a program or application on many connected computers at the same time. One of the main components of the distributed system is the process controller that manages the resources of the system. Load balancing strategy is needed to efficiently utilize the resources among the processors and to minimize the job response time by avoiding a situation where some of the nodes are heavily loaded while other nodes are idle or doing very little work in a distributed cloud environment. To improve the job response time and resource utilization, the proposed system uses the load balancing strategy which includes a Replica model to update the status of main controller after completing each and every job in a distributed environment. Efficient and Enhanced Algorithm, Dynamic Non-Cooperative (DNCOOPC) Algorithm are used for the cloud partitions in a hybrid cloud to minimize the response time and maximize the overall system performance.

KEYWORDS: Load balancing, Distributed Environment, Dynamic Non-Cooperative (DNCOOPC) Algorithm, Efficient and Enhanced Algorithm.

INTRODUCTION

Cloud Computing is visualized as next generation data technology (IT) design for enterprises, attributable to its long list of unexampled benefits within the IT history: on-demand self-service, omnipresent network access, location freelance resource pooling, speedy resource physical property, usage-based rating and transference of risk. Cloud Computing is reworking the terribly nature of how businesses use info technology. “Ref. [4]” Today more modern software development methodologies are being used to enhance the usability of software embedded on compatible hardware in distributed networks. . “Ref. [5],[29]” Middleware characterized as network services and software components that permit scaling of application and networks. Providing the simple and integrated distributed programming environment, middleware eased the task of designing and programming and managing the distributed applications.

Load balancing is emerging technology that facilitates utilization of resource by providing a throughput with minimum response time by sharing the equal load between servers. “Ref. [1]” Best example for load balancing is online shopping cart. Without load balancing, users might expertise delays whereas ordering, transactions and shopping for Load equalization solutions typically applies to redundant servers that provides a more robust distribution of the communication traffic in order that the web buying can be created simple. There are various goals that relates to load balancing strategy such as to improve the performance substantially, to maintain the system stability etc. Depending on the current state of the system, load balancing algorithms can be categorized into two types they are static and dynamic algorithms. In the static algorithm prior knowledge of the system is needed and it does not depends on the current system.

In the case of dynamic algorithm it is based on the current system and it performs better than the static algorithm.[26].

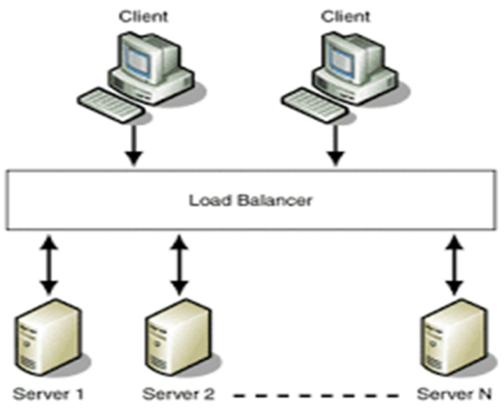

Figure 1. A simple view of load balancer [26]

Dynamic load balancing is a major key for a successful implementation of cloud environments. “Ref. [2]” The main goal of a cloud-based architecture is to provide elasticity, which means the ability of the cloud to expand and contract on demand. Sometimes additional instances of an application will be required in order for the architecture to scale and meet demands. That means there is a necessity for a mechanism to balance requests between two or a lot of instances of that application. The mechanism possibly to achieve success in performing such a task may be a load balancer.

Various algorithms are obtainable for load balancing like Static and Dynamic load equalization. Load balancing within the cloud computing atmosphere has a vital impact on the performance. Sensible load balancing makes cloud computing additional economical and improves user satisfaction. In this research work, we explore Efficient and Enhanced Algorithm and Dynamic Non-Cooperative (DNCOOPC) Algorithm at different situations of load demand in the cloud partitions of a hybrid cloud which manages some resources in-house and has others are provided externally. These dynamic algorithms minimizes the response time and maximizes the overall system performance.

RELATED WORK

Chaczko et al [6] presented “Availability and load balancing in cloud computing” .They recommended that handiness of cloud systems is one amongst the most issues of cloud computing. The term handiness of clouds is principally evaluated by ubiquitousness of data comparison with resource scaling. The research work demonstrated about the critical role of load balancing of the resources improves and maintains the availability of the cloud systems. On this basis the authors proposed an message oriented architecture as a middleware model that improves load balancing in distributed networks and could bring major cost advantages to cloud vendors who are concerned with utility costs and who are searching for efficiencies that can be relatively easily achieved. Ratan et al [7] proposed “Ant colony Optimization: A Solution of Load balancing in Cloud.” Ant colony optimization depicts the load distribution under cloud computing architecture. The pheromone update mechanism is proved to be an effective tool to balance the load. The make span of the cloud computing based services and portability of servicing the request also has been converged using the ant colony optimization technique but this technique failed to address the fault tolerance issues. Kaviani, Nima et al [10] proposed “MANTICORE: A framework for partitioning software services for hybrid cloud.” MANTICORE framework allows code developers to investigate a single host code service to create it suitable for hybrid cloud. In order to verify the accuracy of the cost models, they compared the execution and data transfer measurements of models generated by MANTICORE to those of real deployments for Day Trader. They “Ref. [10] Ashokkumar at al[29]” proposed an extension to existing application partitioning techniques to provide for hybrid deployment of software services. The evaluation on Day Trader showed that the new approach can more effectively contribute towards an optimized hybrid cloud deployment.

Grosu et al [11] proposed “Load balancing in distributed systems: An approach using cooperative games.” The authors developed a static load equalization model for distributed system exploiting cooperative theory of games. It aims to provide optimal allocation scheme. It showed that the Nash Bargaining Solution (NBS) of this game provides a Pareto optimal operation point for the distributed system and it is also a fair solution. The approach also proved that the fairness index is always 1 which means the solution is fair to all jobs.

PROBLEM FORMULATION

The primary purpose of the cloud system is that its consumer will utilize the resources to possess economic advantages. A resource allocation management method is needed to avoid underutilization or over exploitation of the resources which can have an effect on the services of the cloud. A number of incoming jobs is also rejected as the result of overcrowding of the virtual machines within the cloud system. Resource allocation and economical programming are the uncertain characteristic of cloud computing and these factors evaluates the performance of the system. The examined characteristics have a control on value improvement, which may be obtained by improved overall time interval and processing time with the assistance of increased and economical rule. Thus we have proposed a system model which will dynamically balance and migrate the load within the virtual machines to avoid the underutilization and thus enhances the information transfer value.

SYSTEM MODEL

Cloud Deployment

A hybrid cloud is configured which contains a primary main controller, replicated controller, Balancers, and nodes. Hybrid cloud includes many nodes that are situated at different geographical locations. Cloud partitioning is employed to manage the massive cloud design. A cloud partition is sub-area of the hybrid cloud with divisions supported by geographic locations.

Process of Main Controllers and Balancers

After creating the cloud partitions, the load balancing starts initially, the clients provides jobs to the cloud. Primary main controller first communicates with the balancers in each partition and then assigns jobs to the suitable cloud partition .The controller stores the load status information by communicating with the balancers and refreshes the status information at a time period T. Each operations performed by primary main controller was updated to replicated main controller. Balancers in each partition collect the status information from every node and as the job arrives it checks the status of the cloud partition then chooses the right strategy to distribute the jobs. The status may be any one of these: IDLE, NORMAL and HEAVY. If the status is heavy then the jobs gets routed to another partition.

Node Configuration in Cloud Partition

Assigning jobs to the nodes in the cloud partition, Cloud partition balancer gathers load information from every node to evaluate the cloud partition status. First task is to define the load degree node. Each node load degree is associated to various static parameters and dynamic parameters. I.e. Static parameters carries with it the central processing speeds, the amount of CPU's, the memory size, etc and Dynamic parameters area unit the memory utilization magnitude relation, the network information measure, the central processing unit utilization magnitude relation, etc. Load degree results are input into the load status tables created by the cloud partition balancers and then each balancer updates the load status information to the main controller completing each and every task. The status table is then used by the balancers to calculate the partition status. Each partition status incorporates completely different load balancing solution. Once a task enters at a cloud partition, the balancer assigns the task to the node that supports its current load strategy. And this strategy is changed by the balancers as the cloud partition status changes.

- *Compute Load Degree*

Input:

The static parameters include the number of CPU's, the CPU processing speeds, the memory size, etc. Dynamic parameters are the memory utilization ratio, the CPU utilization ratio, the network bandwidth.

Process:-

1. Define a load parameter set: F= {F1,F2…Fm} with each F represents the total number of the parameters.
2. Compute the load degree as Load degree (N) = Σ αiFi Where i= 1…m. αi denotes the weights associated with each job and Fi are the parameters.
3. Average cloud partition degree from the node load degree statistics as:
Load_degree avg =$\sum$ Load_degree (Ni)
4. Three level node status are defined
Load_degree (N) =0 for ***Idle***
0<Load_degree (N) <Load_degree (N) high for ***Normal***
Load_degree (N) high <= Load_degree (N) for ***Overloaded***

Output:-

Idle or Normal or Overloaded

Assigning Jobs to the Cloud Partition

Assign jobs to the cloud, when a job arrives at the hybrid cloud the primary main controller chooses the right partition for the job with the status information provided by the balancer. Cloud partition status can be divided into three types: Idle, Normal, Heavy. The primary main controller has to communicate with the balancers regularly to refresh the status information. **Best partition searching algorithm** is used to assign jobs to the cloud partition based on partition status of nodes in the balancer.

- *Best partition searching algorithm*

```
begin
while job do
```

```
searchBestPartition (job);
if partitionState == idle k partition State == normal then
Send Job to Partition;
else
search for another Partition;
end if
end while
end
```

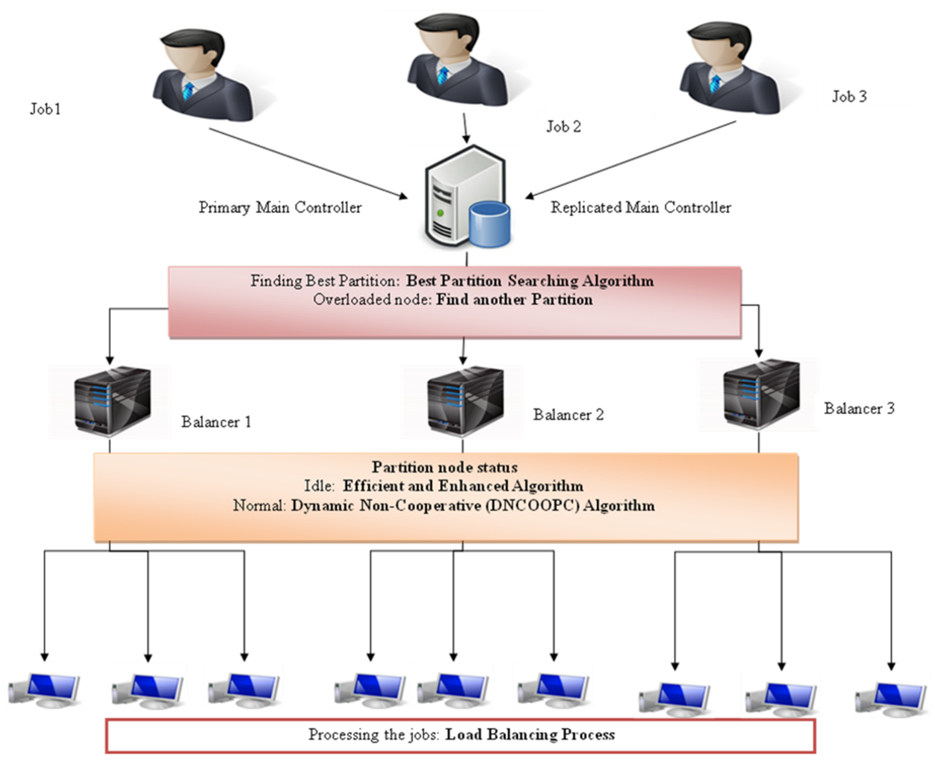
Job1
Job 2
Job 3
Primary Main Controller
Replicated Main Controller
Finding Best Partition: Best Partition Searching Algorithm
Overloaded node: Find another Partition
Balancer 1
Balancer 2
Balancer 3
Partition node status
Idle: Efficient and Enhanced Algorithm
Normal: Dynamic Non-Cooperative (DNCOOPC) Algorithm
Processing the jobs: Load Balancing Process

Figure 2. Basic System Model [27]

Load balancing in Cloud Partition

After assigning the jobs to the cloud partition the balancer uses**, Efficient and Enhanced Algorithm** to process the jobs by calculating expected response time of each node for cloud partition having idle status and **Dynamic Non-Cooperative (DNCOOPC) Algorithm** to process the job by calculating job processing rate, best response rate,and totalload of the node for cloud partition having normal status.

- *Efficient and Enhanced Algorithm (EEA)*

It is a simplest algorithm which can be used in dynamic and distributed load balancing environment.It easily identifies the overall response time and improves the data center processing time as well as cost is reduced in comparison to the existing scheduling parameters.

Step1. Initially VM index table will be 0 as all the VMs are in available state.

Step2. DataCenterController receives a new request.
Step3. DataCenterController queries new LoadBalancer for next allocation
Step4. DataCenterController parses the VM list to get next available VM:
If found: LoadBalancer returns the VM id to DataCenterController continues
If not found:

Using round robin fashion VM index is reinitialised to 0 and in increment manner VMs are checked to find VM in available state

Step5. When the VM finishes the processing the request, and the DataCenterCOntroller recieves the cloulet response, it notices the load balancer of the VM de-allocation
Step6. The Load Balancer updates the status of VM in allocation table to available.
Step7. Continue from Stp2 The purpose of the algorithm is to search out the expected latent period of every Virtual Machine,which is calculated as:
Response Time = Fint - Arrt + TDelay (1)

Where,Arrt is the arrival time of user request and Fint is the finish time of user request and the transmission delay can be determined using the following formulas

TDelay = Tlatency + Ttransfer….(2)

Where, TDelay is the transmission delay T latency is the networklatency and T transfer is the time taken to transfer the size of data of a single request (D) from source location to destination.

Ttransfer = D / Bwperuser (3)

Bwperuser = Bwtotal / Nr (4)

Where, Bwtotal is that the total obtainable information measure and Nr is that the range of user requests presently in transmission. The web Characteristics additionally keeps track of the amount of user requests in between 2 regions for the value of Nr.

- *Dynamic Non-Cooperative algorithm*

we formulate the job allocation problem as a non-cooperative game among the users.
The game theory terminologies are as follows:

Each user $j (j = 1; : : : ; m)$ must find the workload (β) that is assigned to computer i such that the expected price of his own jobs (Dj(β)) is minimized.
The vector βj = [βj1*;* βj2*; : :* βjn] is called the job allocation strategy of user j (j = 1*; : : : ;m*)
The vector β¤ = [*β*1*;β*2*; : : : ; βm*] is called the strategy profile of the job allocation game.
The strategy of user j depends on the job allocation strategies of the other users.

A Non-cooperative job allocation game consists of a collection of players, a collection of strategies, and
a favorable set over the set of strategic profiles:

Players: The m users.
Strategies: Each user's set of feasible job allocation strategies.
Preferences: Each user's preferences are represented by its expected price (Dj). Each user j prefers the strategy profile *β** to the strategy profile *β*'* if and only if *Dj*(*β**) < *Dj*(*β*')*.We need to solve the above game for our job allocation scheme. A solution can be obtained at the Nash equilibrium.
Nash equilibrium: A *Nash equilibrium* of the job allocation game defined above is a strategy profile *β** such that for every user j (j = 1*; : : : ;m*):

$$\beta^{j} \arg\min D_j(\beta_1; : : : ; \beta_j ; : : : ; \beta_m) \qquad (1)$$

The equilibrium strategy profile can be found when each user's job allocation strategy is a *best response* to the other users strategies.The best response for user *j*, is a solution to the subsequent optimization problem (*BRj*):

$$\min_{\beta^{j}} D^{j}(\beta) \qquad (2)$$

subject to the constraints:

$$\beta^{j}_{i} \geq 0; i = 1; : : : ; n \qquad (3)$$

Input:-

Sji be the fraction of jobs that user j send to computer i

The vector sji =(Sj1,Sj2……Sjn) is called the load balancing strategy of user j.

The vector Sj = (S1,S2, ….Sm) is called the strategy profile of the load balancing game

Process :-

1. The expected response time at computer I is

Fi(S) = 1/ µi-Σ j=1 to m sjiɸk

2. The overall expected response time of user j is given by

Dj(S)= Σi= 1 to n sji Fi(S) = Σi=1 to n sji/µi-Σk=1 To m skiɸk

3. The goal of user j is to find a feasible load balancing strategy S ji such that Dj(S)is minimized.

Output:-

The decision of user j depends on the load balancing decisions of other users since Dj(S) is a function of S.

FEATURES OF THE PROPOSED SYSTEM

Cost and time depicts the key challenge of each IT engineer to develop merchandise which will enhance the business performance within the cloud primarily based IT sectors. Current strategies lack economical scheduling and resource allocation techniques resulting in increased operational price and time. This paper aims towards the event of increased ways through improved job programming and resource allocation techniques for overcoming the above-stated problems. Here, *Efficient And Enhanced algorithm* and *Dynamic Non-Cooperative algorithm* dynamically allocates the resources to the job in queue leading reduced price in information transfer and virtual machine formation. Runtime statistics is tracked for every node to adapt to dynamical load needs which makes the system more flexible and compatible with changing user requirements as well as load. Further the failure intensity of the node is also addressed as the system is fault tolerant and balanced. The proposed model works in a dynamically distributed environment which overcomes the drawbacks of single node scheduling decision. The system model also consists of a replicated controller which contains the back up of the processing information and acts as the main controller in case of failure.

COMPARISION OF RESULT

Many of the algorithms are proposed to perform load balancing in various environments. "Ref. [23]" Some of these strategies are compared with the existing model as follows:

Round Robin algorithm provides load scheduling in static environment and uses the method of first come first serve (FCFS) and time quantum. It assigns the time quantum to each node and allocates resources to the client based on the time quantum .Though it works incredibly straightforward however there is an extra load on the hardware to determine the dimensions of quantum and its longer average waiting time, context switches, turnaround and low outturn highly influences the performance of the system. Based on WLC "Ref. [24]" (weighted least connection) algorithm, Ren proposed a load reconciliation technique in dynamic surroundings referred to as ESWLC. It allocates the resource with least weight to a task and takes into consideration node capabilities based on which task is scheduled to a node. LBMM (Load Balancing Min-Min) rule projected in paper " Ref. [25]" uses 3 level frameworks for resource allocation in dynamic surroundings. It uses OLB (opportunistic load balancing) rule as its basis. Since cloud is massively scalable and autonomous, dynamic programming is best choice over static programming. Shagufta proposed load balancing based on Ant Colony Optimization It combine OLB (Opportunistic Load Balancing) to keep each node busy & LBMM(Load Balancing Min Min) to achieve the minimum execution time of each task.

Table 1. Comparison Table of Load Balancing Algorithms in Cloud Computing Environment [28]

ALGORITHM	STATIC ENVIRONMENT	DYNAMIC ENVIRONMENT	CENTRALIZED	DISTRIBUTED
ROUND ROBIN	YES	NO	YES	NO
CLBDM	YES	NO	YES	NO
OLB	YES	NO	YES	NO
ESWLC	NO	YES	YES	NO
ACO	NO	YES	NO	YES
EEA	YES	YES	NO	YES
DNC	YES	YES	NO	YES

CONCLUSION

Proposed system deals with the problem of load balancing in cloud computing with distributed environment. The client sends more number of the tasks to the cloud, where the primary main controller schedules the client's task using best partitioning searching algorithm. After which the task is allocated to load balancers. It assigns the task to the respected node using the dynamic non-cooperative algorithm and efficient and enhanced algorithm. Thus job migration and the dynamic load balancing are achieved in our method. The CPU and Memory is utilized properly and the reliable VM in the cloud pool can be identified. In future different load balancing strategy will be studied because other load balancing strategy may provide better results many tests are needed to compare different strategy which will guarantee better system performance and efficiency.

REFERENCES

K Prasanna Kumar, S.Arun Kumar, Jagadeeshan, Effective Load Balancing For Dynamic Resource Allocation in Cloud Computing, International Journal of Innovative Research in Computer and Communication Engineering Vol. 2, Issue 3, March 2013.

Venubabu, Kunamneni, Dynamic Load Balancing for the Cloud, International Journal of Computer Science and Electrical Engineering (IJCSEE) ISSN No. 2315-4209, Vol-1 Iss-1, 2012.

Load Balancing in the Cloud: Tools, Tips, and Techniques..." http://www.techrepublic.com/resource-library/whitepapers/load-balancing-in-the-cloud-tools-tips-and-techniques/_truncated N.p., n.d. Web. 6 Nov.

R. L. Carter, et al.. Resource allocation in a distributed computing environment, in Digital Avionics Systems Conference, 1998. Proceedings, 17th DASC, AIAA/IEEE/SAE, Vol. 1 (1998), pp. C32/1-C32/8.

R. E. Schantz and D. C. Schidt. Middleware for Distributed System: Evolving The Common Structure for Network-centric Applications. Encyclopedia of Software Engineering. New York, Wiley & Sons (2001), pages801-813.

Availability and Load Balancing in Cloud Computing -IPCSIT." http://ipcsit.com/vol14/25-ICCSM2011-S0063.pdf_truncated. N.p., n.d. Web. 6Nov. 2014

Ratan Mishra, Anant Jaiswal, Ant colony Optimization: A Solution of Load balancing in Cloud, International Journal of Web & Semantic Technology (IJWesT) Vol.3, No.2, April 2012.

K. Nishant, 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 (UKSim), Cambridgeshire, United Kingdom, Mar. 2012, pp. 28-30.

Mayanka Katyal, Atul Mishra, A Comparative Study of Load Balancing Algorithms in Cloud Computing Environment International Journal of Distributed and Cloud Computing, Dec 2013.

Kaviani, Nima, Eric Wohlstadter, and Rodger Lea. "MANTICORE: A framework for partitioning software services for hybrid cloud." In Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on, pp. 333-340, 2012.

D. Grosu, A. T. Chronopoulos, and M. Y. Leung, Load balancing in distributed systems: An approach using cooperative games, in Proc. 16th IEEE Intl. Parallel and Distributed Processing Symp., Florida, USA, Apr. 2002, pp. 52-61.

Rajesh George Rajan, V.Jeyakrishnan, A Survey on Load Balancing in Cloud Computing Environments, International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 12, December 2013.

P. Jamuna and R.Anand Kumar ― Optimized Cloud Partitioning Technique to Simplify Load Balancing‖, International Journal of Advanced Research in Computer Science and Software Engineering, ISSN: 2277 128X, Volume 3, Issue 11, pp. 820 –822, November 2013

Pragathi M, Swapna Addamani, Venkata Ravana Nayak."http://www.ijsrp.org/research-paper-0414/ijsrp-p28103.pdf_truncated. N.p., n.d. Web. 6 Nov. 2014

International Journal of Advanced Research in computer."http://www.ijarcce.com/upload/2013/december/IJ
ARCCE6B-A Rajesh_A_Survey_on_Load_Balancing%20(1).pdf_truncated. N.p., n.d. Web. 6 Nov. 2014

K.Ashokkumar, E.Sankar, Efficient method for parallel process and matching of large data set in grid computing environment ,Journal of Engineering Science and Technology Review, vol 6,2014.

Oğuz Akay ,Kayhan Erciyeş, Dynamic Load Balancing Model For A Distributed System, http:// ube.ege.edu.tr /~erciyes/oguzpaper.doc, aug 2001.

The Future Challenges Of Cloud Computing -Researchomatic.,, http://www.researchomatic.com/The-Future-Challenges-Of-Cloud-Computing-42355.html.

Zenon Chaczko , Venkatesh Mahadevan , Shahrzad Aslanzadeh and Christopher Mcdermid" Availability and Load Balancing in Cloud Computing", 2011 International Conference on Computer and Software ModelingIPCSIT vol.14 (2011) , Singapore.

[20]Load Balancing in the Cloud: Tools, Tips, and Techniques, - http://www.techrepublic.com/resource-library/whitepapers/load-balancing-in-the-cloud-tools-tips-and-techniques/ - RightScale, Apr 2010.

Pragathi M , Swapna Addamani, Venkata Ravana Nayak, " Resource monitoring and workload balancing model for public cloud" , International Journal of Scientific and Research Publications, Volume 4, Issue 4, April 2014.

Suchita Khare, Abhishek Chauhan, " A Review on Load Balancing Model Based on Cloud Partitioning for the Public Cloud", International Journal of Emerging Technology and Advanced Engineering, Volume 4, Issue 7, July 2014.

Mayanka Katyal , Atul Mishra , "A Comparative Study of Load Balancing Algorithms in Cloud Computing Environment", http://www.publishingindia.com.

Lee, R. & Jeng, B. (2011). Load-balancing tactics in cloud. In proc. International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), IEEE, (pp. 447-454).

Wang, S. C., Yan, K. Q., Liao, W. P. & Wang, S.S. (2010). Towards a load balancing in a threelevelcloud computing network. Proceedings of 3rd International Conference on Computer Science and Information Technology (ICCSIT), IEEE, July, 1,108-113.

http://www.2bnet.co.il/63

JC Juan Chen," Dynamic Spectrum Load Balancing for Cognitive Radio in Frequency Domain and Time Domain", The Journal of The Korea Institute of Intelligent Transport Systems,2009. http://www.earticle.net/article.aspx?sn=111349.

Mayanka Katyal*, Atul Mishra, "A Comparative Study of Load Balancing Algorithms in Cloud Computing Environment", nternational Journal of Distributed and Cloud Computing, Volume 1 Issue 2 December 2013.

K.Ashokkumar, C.chandrasekar, "Data management and heterogeneous data integration in Grid computing environment", proceedings of the 2010 International Conference on Communication and Computational Intelligence (INCOCCI 2010), IEEE Catalog Number: CFP1040N-PRT , ISBN: 9781457703768.