ISSN:0975-9646

Reviews of Load Balancing Based on Partitioning in Cloud Computing

Ms.Shilpa D.More[1], Mrs.Smita Chaudhari[2]

[1,2] *Department of Computer Engineering,*
DYPSOET, Pune, India.

***Abstract*- Load Balancing Model Based on Cloud Partitioning for the Public Cloud environment has an important impact on the performance of network load. A cloud computing system which does not use load balancing has numerous drawbacks. Now-a-days the usage of internet and related resources has increased widely. Due to this there is tremendous increase in workload. So there is uneven distribution of this workload which results in server overloading and may crash. In such systems the resources are not optimally used. Due to this the performance degrades and efficiency reduces. Cloud computing efficient and improves user satisfaction. This article introduces a better load balance model for public cloud based on the cloud partitioning concept with a switch mechanism to choose different strategies for different situations. The algorithm applies the game theory for load balancing strategy to improve the efficiency in the public cloud environment.**

Key words: **load balancing model, public cloud, cloud partition, game theory.**

I. INTRODUCTION

Cloud Computing is a concept that has many computers interconnected through a real time network like internet. cloud computing means distributed computing. Cloud computing enables convenient, on-demand, dynamic and reliable use of distributed computing resources. The cloud computing model has five main characteristics on demand service, broad network access, resource pooling, flexibility, measured service.

Cloud computing is efficient and scalable but to maintain the stability of processing many jobs in the cloud computing is a very difficult problem. The job arrival pattern cannot be predicted and the capacities of each node in the cloud differ. Hence for balancing the usage of internet and related resources has increased widely. Due to this there is tremendous increase in workload. So there is uneven distribution of this workload which results in server overloading and may crash. In such the load, it is crucial to control workloads to improve system performance and maintain stability. The load on every cloud is variable and dependent on various factors. To handle this problem of imbalance of load on clouds and to increase its working efficiency, this paper tries to implement "A Model for load balancing by Partitioning the Public Cloud". Good load balancing makes cloud computing more efficient and also improves user satisfaction [1]. This article is aimed at the public cloud which has numerous nodes. A system having main controller, balancers, servers and a client is implemented here. It introduces a switch mechanism to choose different strategies for different situations. This paper divides the public cloud into cloud partitions and applies different strategies to balance the load on cloud.

This paper gives an idea for balancing the load on clouds. It helps to avoid overloading of servers and improve response times. The basic designs of the system and algorithms to implement it are described in this paper [1]

Goals of Load Balancing

- To improve the performance substantially.
- To have a backup plan in case the system fails even partially.
- To maintain the system stability.
- To accommodate future modification in the system.

II. LATEST ISSUES

The jobs arrival pattern is not predictable and the capacities of each node in the cloud differ, for the load balancing problem, and workload control is difficult to improve system performance and maintain stability. Cloud computing is efficient and

scalable but maintaining the stability of processing so many jobs in the cloud computing environment is a very complex problem with load balancing receiving much attention for researchers.

.

III. PROPOSED ISSUES

(1) Cloud division rules: Cloud division is not a simple problem. Thus, the framework needs a detailed cloud division methodology. For example, nodes in a cluster may be far from other nodes or there will be some clusters in the same geographic area that are still far apart. The division rule should simply be based on the geographic location.

(2) How to set the refresh period for data statistics analysis, the main controller and the cloud partition balancers need to refresh the information at a fixed period. If the period is too short, the high frequency will influence the system performance. If the period is too much long, the information will be too old to make good decision. Thus, tests and statistical tools are needed to set reasonable refresh periods.

(3) A load status evaluation: A good algorithm is needed to set Load degree high and Load degree low, and the evaluation mechanism needs to be more comprehensive.

(4) Find out other load balance strategy: and other load balance strategies may provide better results, so tests are needed to compare different strategies. Many tests are needed to guarantee system availability and efficiency

IV. RELATED WORK

There have been many studies of load balancing for the cloud environment. Load balancing in cloud computing was described in a white paper written by Adler[3] who introduced the tools and techniques commonly used for IEEE TRANSACTIONS ON CLOUD COMPUTING YEAR 2013 load balancing in the cloud. However, load balancing in cloud is still a new problem that needs new architectures to adapt too many changes. Chaczko et al.[4] described the role that load balancing plays to improve the performance and maintaining stability.

There are many load balancing algorithms, such as Round Robin, Game theory Algorithm, and Ant Colony algorithm. Nishant et al.[5] used the ant colony optimization methods in nodes load balancing. Randles et al.[2] gave a compared analysis of some of algorithms in cloud computing by checking the performance time and cost. They concluded that the ESCE algorithm and throttled algorithm are better than Round Robin algorithm. Some of the classical load balancing methods is similar to the allocation method in the operating system, for example, the Round Robin algorithm and the First Come First Served (FCFS) rules. The Round Robin algorithm is used here because it is fairly simple.

V. LOAD BALANCING

In cloud computing, load balancing is required to distribute the dynamic local workload evenly across all the nodes. It helps to achieve a high user satisfaction and resource utilization ratio by ensuring an efficient and fair allocation of every computing resource. Proper load balancing aids in minimizing resource consumption, implementing fail-over, enabling scalability, avoiding bottlenecks and over-provisioning.

There are mainly two types of load balancing algorithms: In static algorithm the traffic is divided evenly among the servers. This algorithm requires a 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 is proper in the system which has low variation in load. In dynamic algorithm the lightest server in the whole network or system is searched and preferred for balancing a load. For this real time communication with network is needed which can increase the traffic in the system. Here current state of the system is used to make decisions to manage the load.

Load balancing based on Cloud Partitioning

There are several cloud computing services with this work focused on a public cloud. A public cloud is based on the standard cloud computing model, with service provided by a service provider. A large public cloud will include many nodes and the nodes in different geographical locations. Cloud partitioning is used to manage this large cloud. A cloud partition is a subarea of the public cloud with divisions based on the geographic locations. The architecture is shown in Fig.1. The load balancing strategy is based on the cloud partitioning concept. After creating the cloud partitions, the load balancing then starts, when a job arrives at the system, with the main controller deciding which cloud partition should receive the job. The partition load balancer then decides how to assign the jobs based on load status of nodes. When the load status of a cloud partition is normal, this partitioning can be accomplished locally. If the cloud partition load status is not normal, this job should be transferred to another partition.

Here we are going to discuss some load balancing technique for both the partition having either load status=idle or load status=normal based on load degree. The node load degree is based on different static and dynamic parameters of each node.

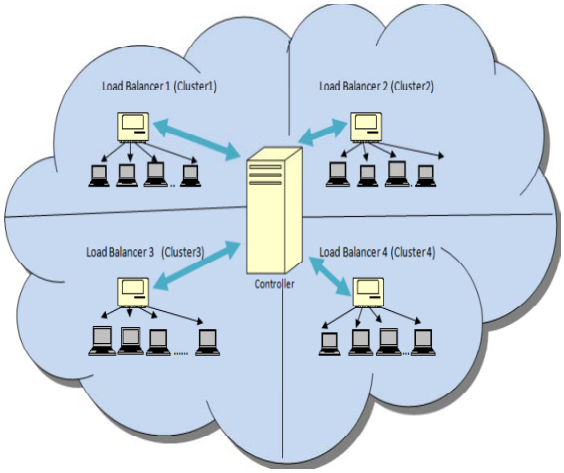

Fig 1 :- Load balancing Architecture

For cloud partition having idle status:

In this situation, this cloud partition has the ability to process jobs as quickly as possible so a simple load balancing method can be used. There are lots of works has been done for load balance algorithm such as the Random algorithm, the Weight Round Robin, and the Dynamic Round Robin[2]. The Round Robin (RR) is used here because it is very simple method for load balancing. The Round Robin algorithm does not record the status of each connection so it has no status information. In a public cloud, the configuration and the performance of each node will be not the same; thus, this method may overload some nodes. Thus, an improved Round Robin algorithm is used, which called "Round Robin based on the load degree evaluation". Before the Round Robin step, the nodes in the load balancing table are ordered based on the load degree from the lowest to the highest. The system builds a circular queue and walks through the queue again and again. Jobs will then be assigned to nodes with low load degrees. The node order will be changed when the balancer refreshes the Load Status Table.

For cloud partition having Normal status:

This situation is more complex than the idle status situation, because in these situations jobs are dispatched faster by the cloud Load Balancer and each user wants to execute his job at shortest response time so the public cloud. S. Penmatsa and Chronopoulos [8] has proposed

"static load balancing strategy based on game theory for distributed systems". This paper is the base of our review work and we consider that the implementation of distributed system, the public cloud load balancing can be viewed as a game. The purpose of load balancing is to improve the performance of a system through an appropriate distribution of the application load. A general formulation of this problem is as follows: given a large number of jobs, find the allocation of jobs to computers optimizing a given objective.

"Game theory is the formal study of decision-making where several players must make choices that potentially affect the interests of the other players". Game theory is the formal study of conflict and cooperation. Game theoretic concepts apply whenever the actions of several agents are interdependent. These agents may be individuals, groups, firms, or any combination of these. The concepts of game theory provide a language to formulate structure, analyze, and understand strategic scenarios.

A game is a description of strategic interaction that includes the constraints on the actions that the players can take and the players' interests, but does not specify the actions that the players do take. A solution is a systematic description of the outcomes that may emerge in a family of games. Game theory suggests reasonable solutions for classes of games and examines their properties. There are three types of game theocratic load balancing techniques for public cloud.

Global Method – In this case there is only one decision maker that optimizes the response time of the entire system over all jobs and the operating point is called social (overall) optimum

Cooperative Method – In this case there are several decision makers (e.g. jobs, computers) that cooperate in making the decisions such that each of them will operate at its optimum. Decision makers have complete freedom of pre-play communication to make joint agreements about their operating points.

Non cooperative Method – In this case there are several decision makers (e.g. users, jobs) that are not allowed to cooperate in making decisions. Each decision maker optimizes its own response time independently of the others and they all eventually reach equilibrium. This situation can be viewed as a noncooperative game among decision makers[6]

The equilibrium is called Nash equilibrium and it can be obtained by a distributed non cooperative policy. At the Nash equilibrium a decision maker cannot receive any further benefit by changing its own decision. In the game of load balancing for the public cloud the players would be nodes in each cloud partition and the user jobs dispatched by the Load Balancer. We assume that there are n nodes in each partition and p jobs dispatched by the LB. Now the Load Balancer (LB) has to decide on how to distribute user jobs to available nodes such that they will operate optimally. In the following, we present the notations we use and then define the non-cooperative load balancing game

μi :- Average Processing Time of node where node i=1, 2, 3…… n

Øj;- Time spending of each job where job j=1, 2, 3….. m

Φ :- ∑ Øj, is the Total Time spent by cloud partition

Thus user j (j=1, 2, 3,……m) must find the fraction Sji of all its jobs that are assigned to the node i such that expected execution time of this job is minimized. Let us assume that Sji is the fraction of job j is assigned to node i. The vector Sj=(Sj1, Sj2, …….. Sjn) is called the load balancing strategy of user job j. And the vector S= (S1, S2, …….. Sn) is called the strategy profile of load balancing game. From that workload we are calculating overall expected response time of user and node.

VI. CONCLUSION

The overall goal of this project is to balance the load on clouds. Balancing load on the cloud will improve the performance of cloud services substantially. It will prevent overloading of servers, which would otherwise degrade the performance. The response time will also improve.

This software maybe used for efficient data storage on clouds and load balancing. This software will help dynamically allocate jobs (data) to the least loaded server. Thus overall performance of cloud services will not be affected.

It aims at having a backup plan in case the system fails even partially. Also work is done to maintain the system stability. There are provisions to accommodate future modifications in the system. Thus, we have successfully gathered information of project and hopefully implement Load Balancing Model for better utilization and performance of cloud services.

ACKNOWLEDGEMENT

The authors would like to thank to Gaochao Xu, Junjie Pang, and Xiaodong Fu for their exclusive information and valuable comments.

REFERENCES

1. Gaochao Xu, Junjie Pang, and Xiaodong Fu* "A Load Balancing Model Based on Cloud Partitioning for the Public Cloud ", ISSN l1007-0214l l04/12l lpp34-39,Volume 18, Number 1, February 2013.
2. M. Randles, D. Lamb, and A. Taleb-Bendiab, A comparative study into distributed load balancing algorithms for cloud computing, in Proc. IEEE 24th International Conference on Advanced Information Networking and Applications, Perth, Australia, 2010.
3. B. Adler, Load balancing in the cloud: Tools, tips and Tools, tips and techniques, http://www.rightscale. com/info center/whitepapers/ Load-Balancing-in-the-Cloud.pdf, 2012
4. Z. Chaczko, Venkatesh Mahadevan, Shahrzad Aslanzadeh, Christopher Mcdermid. "Availability and Load Balancing in Cloud Computing" 2011 International Conference on Computer and Software Modeling.
5. 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 Modeling and Simulation (UKSim), Cambridgeshire, United Kingdom, Mar. 2012, pp. 28-30.
6. D. Grosu, A. T. Chronopoulos, Non cooperative Load balancing in distributed system , Journal of Parallel and Distributed computing ,2005
7. P. Mell and T. Grance, The NIST definition of cloud computing, http://csrc.nist.gov/ publications/nist pubs/800-145/SP800-145.pdf, 2012.
8. S. Penmatsa and A. T. Chronopoulos, Game-theoretic static load balancing for distributed systems, Journal of Parallel and Distributed Computing, vol. 71, no. 4, pp. 537-555, Apr. 2011.