Balancing Server in Public Cloud Using AJAS Algorithm

Ramya.B[1],
M.Phil Part-Time Research Scholar,
Department of Computer Science,
Sri Vasavi College,
Erode, India[1].

Pragaladan R[2],
Assistant Professor
Department of Computer Science,
Sri Vasavi College,
Erode, India[2].

Abstract-. Cloud computing is an attracting technology in the field of computer science. It is proven that cloud will bring changes to the IT industry. The cloud is changing our life by providing users with new types of services. Users get service from a cloud without paying attention to the details. In order to avoid the traffic and unwanted mechanism of threat in require a perfect load balancing. Load balancing in the cloud computing environment has an important impact on the performance. The main aim of Good load balancing makes cloud computing more efficient and improves user satisfaction. This project introduces a better load balance model for the public cloud based on the cloud partitioning concept with a switch mechanism to choose different strategies for different situations. The algorithm applies particular methods the game theory and AJAS algorithm (Adaptive Scoring Job Scheduling Algorithm) to the load balancing strategy to improve the efficiency in the public cloud environment

Keywords - ***Cloud Computing, Public Enterprise, Load Balancing, AJAS.***

I. INTRODUCTION

Cloud computing portends a major change in how we store information and run applications. Instead of running programs and data on an individual desktop computer, everything is hosted in the "cloud"—a nebulous assemblage of computers and servers accessed via the Internet. Cloud computing lets you access all your applications and documents from anywhere in the world, freeing you from the confines of the desktop and making it easier for group members in different locations to collaborate.

Cloud Computing Technology is perceived by many as a new asset of Information technology for the IT companies, educational institutions, government sectors, etc. In the ever fast growing economy apart from the challenges faced due to recession, the educational institutes find this a big hurdle as to how to provide necessary Information technology support for educational activities and research areas. Cloud Computing, the latest buzzword in IT sector, may come to the rescue, as it can provide an easy and inexpensive access to the state of the art IT technology, software and its applications. Cloud computing is a recent concept that is still evolving across the information technology industry and academia.

Cloud computing is Internet (cloud) based development and use of computer technology whereby dynamically scalable and often virtualized resources are provided as a service over the Internet. This research paper aims at studying the factors which affect the adoption of Cloud Computing Technology in an educational institution, case study of a Delhi based renowned public school. Questionnaire was used a data collection tool and the results were analyzed by SPSS program for statistical analysis.

II.SERVICE UNDER LOAD BALANCING ENVIRONMENT

Cloud computing contains many content like virtualization, distributed computing, networking, advance version to grid computing, software and web services, etc. A cloud consists of components like Cloud controller, cluster controller, nodes, datacenters and servers. It composed of fault tolerance, high availability, scalability, flexibility, reduced cost of ownership, reduced overhead for users, less maintainability, on demand services. Addressing to this issue needs the foundation of efficient load balancing algorithm.

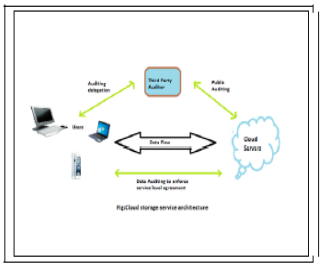


Fig-1 Cloud Computing Basic Storage Structure

The load can be CPU load, memory capacity, delay or network load. Load balancing is the process of distributing the load among various nodes of a distributed system to improve both resource utilization and job response time

while 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 the processor in the system or every node in the network does approximately the equal amount of work at any instant of time. Various types of cloud system

• **Platform as a Service (PaaS):** Users create and run their own software applications while relying on the cloud provider for software development tools as well as the underlying infrastructure and operating system.

• **Software as a Service (Saas):** Under this layer applications are delivered through the medium of the internet as a service. Instead of installing and maintaining software, you simply access it via the internet, freeing yourself from complex software and hardware management. The cloud provider hosts a single application which offers complete application functionality.

• **Infrastructure as a Service (IaaS):** Users rent computing power – either actual hardware or virtualized machines – to deploy and run their own operating systems and software applications. Simply, the backend systems that deliver cloud services are generally deployed in one of four ways.

• **Public Cloud:** Customers access cloud services are store documents in large documenters Equipped with hundreds of virtualized servers that house data from multiple organizations.

• **Private Cloud:** A single organization uses a dedicated cloud infrastructure. Community cloud: A private cloud is shared by a group of organizations with common missions, interests, or concerns.

• **Hybrid cloud:** Two or more cloud types are linked to enable data and applications to flow between them in a controlled way.

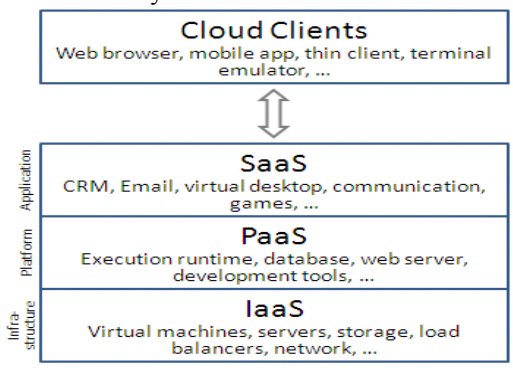


Fig-2 Cloud Computing Basic Types Structure

III.PROBLEM IDENTIFIED

The system introduces a better load balance model for the public cloud based on the cloud partitioning concept with a switch mechanism to choose different strategies for different situations.

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 to the 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.

The system develops a resource allocation system that can avoid overload in the system effectively while minimizing the number of servers used. It introduces the concept of "skewness" to measure the uneven utilization of a server. By minimizing skewness, it can improve the overall utilization of servers in the face of multidimensional resource constraints.

It designs a load prediction algorithm that can capture the future resource usages of applications accurately. The algorithm can capture the rising trend of resource usage patterns and help reduce the placement churn significantly.It looks inside a VM for application level statistics, e.g., by parsing logs of pending requests. Doing so requires modification of the VM which may not always be possible.

Demerits Of System In Public Cloud

The following drawbacks are present in the system.

- The mechanism is calculating cloud area status only after the job is completed.Cloud areas are grouped based on geographical locations only.
- All nodes inside the cloud area are treated equally irrespective of their capabilities.
- One node is assigned with the given job.
- Cloud Node failure scenario is not taken into account.
- All jobs are independent during scheduling and then assigned to cloud nodes.
- Job replication strategy (such as upload or search process) is not considered.
- Job completion time is not reduced.

IV.SYSTEM MODEL

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 load balancing

strategy is based on the cloud partitioning concept. After creating the cloud partitions, the load balancing then starts: when a job arrives . Typical cloud partitioning. 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 to the nodes. When the load status of a cloud partition is normal, this partitioning can be accomplished locally. If he cloud partition load status is not normal, this job should be transferred to another partition.

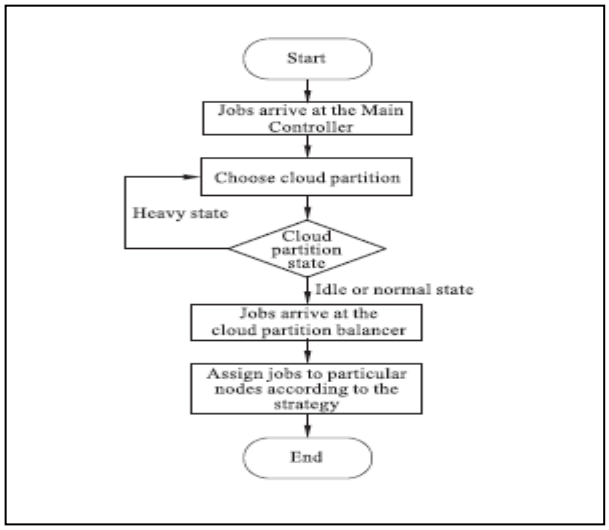


Fig-3 Cloud Computing Structure of load balancing

1) Addition Of Area

In addition of area, the area's Id, area name and bandwidth details are keyed in and saved in 'Area's table. The details are viewed using data grid view control. Those area id, name and bandwidth details are entered in the text box control. The user adds, saves, and updates the details in the Area table.

2) Addition Of Node

In aition of node, the node's Id, node name, IP address, area name, RAM capacity, Total storage capacity, balance storage capacity, and CPUMHz details are keyed in and saved in 'Nodes's table. The details are viewed using data grid view control. Those node's Id, node name, IP address, area name, RAM capacity, total storage capacity, balance storage capacity, and CPUMHz details are entered in the text box control.

3) Addition Of Job

In addition of job, the Job Id, job name, Require RAM, Require storage, CPU MHz and bandwidth details are keyed in and saved in 'Job's table. The details are viewed using data grid view control. Those the Job Id, job name, Require RAM, Require storage, CPU MHz and bandwidth details are entered in the text box control. The user can add, save, and update the details in the jobs table.

4) Addition Of Load Balancer

In addition of load balancer, the area's Id, balancing node id details are keyed in and saved in 'load balancer's table. The details are viewed using data grid view control. Those area's Id, balancing node id details are entered in the text box control. The user can add, save, and update the details in the Area table.

5) Assign Job

In assigning of the job Id, job name, require RAM, require storage, CPU MHz and bandwidth details are keyed in and saved in 'Job's table. The details are viewed using data grid view control. Those the Job Id, job name, Require RAM, Require storage, CPU MHz and bandwidth details are entered in the text box control. Based on the job table details the job will be assign to the particular area which is capable for the assigned job.

6) Assign Job (Split Strategy)

In assigning of job in split sequence, the job Id, job name, require RAM, require storage, CPU MHz and bandwidth details are keyed in and saved in 'Job's table. The details are viewed using data grid view control. Those the Job Id, job name, Require RAM, Require storage, CPU MHz and bandwidth details are entered in the text box control. Based on the job table details the job will be split based on the RAM, storage, CPU and bandwidth details and those multiple sub-jobs are assigned one by one to the particular areas which are capable for the assigned job.

7) Skewness Algorithm Execution

a) Addition of Node

In node addition, the node's IP Address and system name are keyed in and saved in 'Node's table. The details are viewed using data grid view control.

b) Prepare 'CPU' and 'Disk Transfer' Data

A windows application is designed which acts as grid data collector. The application collects the grid usage data such as CPU usage, memory statistics are stored in a file. The CPU usage is retrieved at specified intervals say 30 seconds and kept IN buffer. At regular interval they are stored in a log file in same node. During upload, they are converted to XML files and are submitted to the server database so that they can be consolidated in a common format.

c) Consolidate 'CPU' and 'Disk Transfer' Data

Upon regular schedule, the collected data are converted to XML files and are submitted to the server database so that they can be consolidated in a common format. The statistics of usage data can be viewed in data grid view control as well as chart control for easy analysis. During chart preparation, individual usages such as CPU usage, disk usages are prepared for each node or selected nodes or all nodes.

d) Server Skewness Calculation

In this module, the concept of skewness is introduced to quantify the unevenness in the utilization of multiple resources on a server. Let **n** be the number of resources and **ri** be the utilization of the **i**th resource. The resource skewness of a server **p** is defined as

$$skewness(p) = \sqrt{\sum_{i=1}^{n}\left(\frac{r_i}{\overline{r}} - 1\right)^2},$$

where $\overline{\mathbf{r}}$ is the average utilization of all resources for server **p**. In practice, not all types of resources are performance critical and hence we only need to consider bottleneck resources in the above calculation. By minimizing the skewness, different types of workloads are combined nicely and the overall utilization of server resources is improved.

8) Hot And Cold Spots

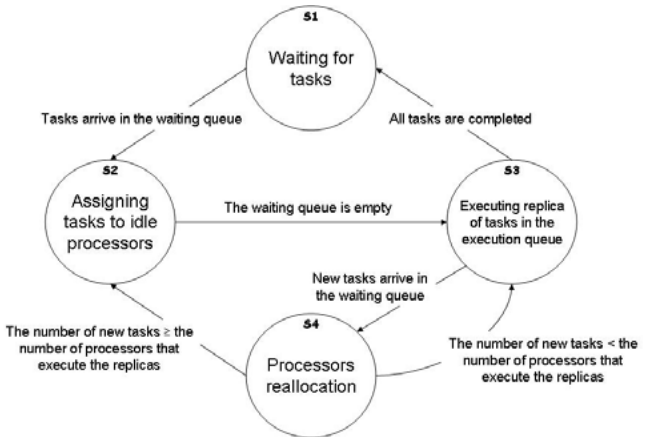
The algorithm executes periodically to evaluate the resource allocation status based on the predicted future resource demands of VMs. It defines a server as a hot spot if the utilization of any of its resources is above a hot threshold. This indicates that the server is overloaded and hence some VMs running on it should be migrated away. It defines the temperature of a hot spot p as the square sum of its resource utilization beyond the hot threshold:

$$temperature(p) = \sum_{r \in R}(r - r_t)^2,$$

where **R** is the set of overloaded resources in server **p** and **rt** is the hot threshold for resource **r**. (Note that only overloaded resources are considered in the calculation.) The temperature of a hot spot reflects its degree of overload. If a server is not a hot spot, its temperature is zero.

9) Adaptive Scoring Job Scheduling Algorithm

In among all the cloud nodes, Adaptive Scoring Job Scheduling algorithm (ASJS) is applied for cloud nodes resource scheduling so that the given job is split into 'N' tasks along with Replication Strategy.



State I:
Initial phase: State I is represented by an idle scheduler waiting for tasks to arrive in the grid. Incoming tasks are lined up in the waiting queue. Both the Waiting Queue and the Execution are initially empty. When the number of tasks in the Waiting Queue becomes more than the threshold value, a transition is made to State II.

State II:
Initial phase: The Execution Queue is initially empty while the Waiting Queue comprises of a number of incoming tasks in the Grid. We maintain a list comprising of idle machines in the system. Initially all the machines are idle in State II, and hence the list contains all the machines in the Grid. The tasks from the head of the Waiting queue and mapped one by one to the machines in the idle list. As soon as a task from the Waiting queue gets mapped to a machine, the given machine is subsequently removed from the idle list, while the task is removed from the head of the Waiting Queue and inserted into the Execution Queue as explained in the System Model. The logic behind the mode of insertion is to give the highest priority to a task which has been assigned the slowest processor and vice-versa. The task with the highest priority in the Execution Queue would

be the first one to get replicated because it is essentially the task with the slowest processors dedicated to it and hence replicating such a task would lead to a very high probability of the new machine executing the task before the machines already assigned to it.

State III:
Initial phase: The waiting Queue is initially empty while the Execution Queue consists of tasks that are in execution in the Grid.If a machine completes the execution of a task, the processor list of that particular task is referred to, and all the machines dedicated to executing the given task are released and made free while the task is removed the Execution Queue and the current, next and last pointers are updated if required. We update the processing power of the newly freed machine based upon the number of instructions that the machine executed for the previous task and the time it took to finish its execution. If the processing speed of the machine is greater than that of the *maxProcSpeed* of the task pointed to by the current pointer, then the given machine is required to execute the replica of the task pointed to by the current pointer. Also, the current, next and last pointers are updated as follows. The current pointer becomes the last pointer, the next pointer becomes the current pointer and the next pointer would now be pointing to the task that was one step in the clockwise direction to the task pointed to by the erstwhile next pointer. Also, if the machine assigned to execute the replica of a task has a processor speed greater than that of *maxProcSpeed* of that task then the value needs to be updated to the processing speed of the given machine. We also keep track of the number of machines executing replicas and the number of tasks in the Waiting Queue, the information of which is exploited in StateIV of the heuristic. At this point of time, we also check there are three scenarios, eventually possible, in State III.

Case I*:* All the tasks in the Execution Queue are successfully completed while the Waiting Queue is still empty. In this case, we traverse back to State I.

Case II*:* All the tasks in the Execution Queue are successfully completed while the number of tasks in the Waiting Queue is still less or equal to the threshold. In this case, we traverse back to State II.

Case III: The number of tasks in the Waiting Queue has exceeded the threshold before all the tasks in the Execution Queue could be completed. In this case, we traverse to State IV.

State IV
Initial Phase: Both the Waiting and the Execution Queue are initially non-empty. Two scenarios are possible at this point of time.

Case I: The number of machines executing replicas is less than the number of tasks in the Waiting Queue.

Case II: The number of machines executing task replicas is more than the number of tasks in the Waiting Queue. The tasks in the Execution Queue are traversed in an anticlockwise manner one by one, starting from the task pointed to by the last pointer (the least priority task) and if a task has more than one machine allocated to it, the machine at the tail end of the processor list is taken out of the list, freed from the task it was currently executing and assigned the task at the head of the Waiting Queue (after removing the task from the queue).

In Case I, we stop the traversal as soon as all the tasks in the execution queue are being run on one and only one machine an transition to State II. In Case II, we stop traversing the linked list as soon as all the machines in the Waiting Queue are assigned a machine, and then subsequently transition to State III.

V. CONCLUSION

In system multiplexes virtual to physical resources adaptively based on the changing demand. We use the skewness metric to combine VMs with different resource characteristics appropriately so that the capacities of servers are well utilized. Our algorithm achieves both overload avoidance and green computing for systems with multiresource constraints.

Cloud division is not a simple problem. Thus, the framework will need 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 (province or state). In the 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 long, the information will be too old to make good decision. Thus, tests and statistical tools are needed to set a reasonable period.

VI. FUTURE SCOPE

The following are to be present in the proposed system.

- The new mechanism is calculating cloud area status at a given refresh period.
- Cloud areas are grouped based on their processing ability.
- Nodes inside the cloud area not treated equally. According to their processing and storage power, the partial job is assigned to them.
- One job is assigned to multiple node after the job is split according to the nodes capability.
- Jobs are split into sub tasks and assigned to more cloud nodes.
- Both dependent tasks and independent task scheduling is taken into account.
- Job replication strategy is also considered.
- Decreasing completion time of jobs.

REFERENCE

[1] M. Armbrust et al., "Above the Clouds: A Berkeley View of Cloud Computing," technical report, Univ. of California, Berkeley, Feb. 2009.

[2] L. Siegele, "Let It Rise: A Special Report on Corporate IT," The Economist, vol. 389, pp. 3-16, Oct. 2008.

[3] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R.Neugebauer, I. Pratt, and A. Warfield, "Xen and the Art of Virtualization," Proc. ACM Symp. Operating Systems Principles (SOSP '03), Oct. 2003.

[4] "Amazon elastic compute cloud (Amazon EC2)," http://aws.amazon.com/ec2/, 2012.

[5] C. Clark, K. Fraser, S. Hand, J.G. Hansen, E. Jul, C. Limpach, I.Pratt, and A. Warfield, "Live Migration of Virtual Machines,"Proc. Symp. Networked Systems Design and Implementation (NSDI '05), May 2005.

[6] M. Nelson, B.-H. Lim, and G. Hutchins, "Fast Transparent Migration for Virtual Machines," Proc. USENIX Ann. Technical Conf., 2005.

[7] M. McNett, D. Gupta, A. Vahdat, and G.M. Voelker, "Usher: An Extensible Framework for Managing Clusters of Virtual Machines," Proc. Large Installation System Administration Conf. (LISA '07), Nov. 2007

[8] T. Wood, P. Shenoy, A. Venkataramani, and M. Yousif, "Black-Box and Gray-Box Strategies for Virtual Machine Migration," Proc. Symp. Networked Systems Design and Implementation (NSDI '07), Apr. 2007.

[9] C.A. Waldspurger, "Memory Resource Management in VMware ESX Server," Proc. Symp. Operating Systems Design and Implementation (OSDI '02), Aug. 2002.

[10] G. Chen, H. Wenbo, J. Liu, S. Nath, L. Rigas, L. Xiao, and F. Zhao, "Energy-Aware Server Provisioning and Load Dispatching for Connection-Intensive Internet Services," Proc. USENIX Symp. Networked Systems Design and Implementation (NSDI '08), Apr. 2008.

BIOGRAPHY

Mr.R.Pragaladan received his B.Sc and M.Sc degree in Computer Science from the Bharathidasan University in 2003 and 2006 respectively. He received his M.Phil degree in Computer Science from Bharathidasan University in the year 2007. He is currently pursuing Ph.D in Computer Science from Bharathiar University. He is presently working as an Assistant Professor, Department Of Computer Science, Sri Vasavi College, Erode – 638316. He has 6 years Teaching and Research Experience. His Research Areas are Cloud Computing, Network Security and Mobile Computing.

Ms. Ramya B is Part-Time research scholar pursuing her M. Phil in the area of Computer Science in Sri Vasavi College, Erode affiliated by Bharathiyar University. She has also completed her Post Graduation in Computer Technology in Institute Kongu Engineering College in Erode affiliated by Anna University. Her Research areas are Cloud Computing, Grid Computing and Networking.

