ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 2, pp: (632-637), Month: April - June 2015, Available at: www.researchpublish.com

A Dynamic Load Balance Model for Partitioned Public Cloud

Sowmya Shree P[1], Mr. M Jayashankar[2]

PG Student (CS&E) [1], Asst. Professor (Dept of CS&E) [2]
P.E.S. College of Engineering, Mandya, India,
An Autonomous Institution under Visvesvaraya Technological University, Belgaum, India

Abstract: **Cloud computing is a major paradigm in the technology world. Load balancing has the major impact on the performance of the cloud. Thus, an effective load balancing plays an important role in the different aspects of the cloud. Even though there are several load balancing techniques available, the article provides a different way of load balancing based on the technique of partitioning of cloud based on geographical locations and a switch mechanism is used to select the partitions. The technique uses Hungarian method of assignment problem to improve the efficiency of the load balancing technique in the public cloud environment.**

Keywords: **Load balancing; Public cloud; Cloud partition; Hungarian method; Assignment problem.**

1. INTRODUCTION

Cloud computing is a new paradigm in which computing resources such as processing, memory, and storage are not physically present at the user’s location. Instead, a service provider owns and manages these resources, and user accesses them via the Internet. 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.

Cloud computing is provides hosted services over the Internet. These services are broadly divided into three categories: *Infrastructure-as-a-Service (IaaS), Platform-as-a- Service (PaaS)* and *Software-as-a-Service (SaaS)* [1]. A cloud can be private or public. A public cloud sells services to anyone on the Internet. A private cloud is a proprietary network or a data centre that supplies hosted services to a limited number of people. When a service provider uses public cloud resources to create their private cloud, the result is called a virtual private cloud. Private or public, the goal of cloud computing is to provide easy, scalable access to computing resources and IT services.

Cloud computing is the use of computing resources that are delivered as a service over a Load balancing and provisioning in cloud computing systems is really a challenge job. For solving such problem always a distributed and dynamic solution is required. Because it is not always practically feasible or cost efficient to maintain one or more idle services just as to fulfils the required demands jobs cannot be assigned to appropriate servers and clients individually for efficient load balancing as cloud is a very complex structure and components are present throughout a wide spread area. Here, some uncertainty is attached while jobs are assigned. Public cloud services may be free or offered on a pay-per-usage model. The term "public cloud" arose to differentiate between the standard model and the private cloud, which is a proprietary network or data canter that uses cloud computing technologies, such as virtualization. Examples of public clouds include Amazon Elastic Compute Cloud (EC2), IBM's Blue Cloud, Sun Cloud, Google Drive, Google Apps, and Windows Azure Services Platform [1]. Here, a dynamic strategy to balance workload among nodes with the help of cloud partitioning is reviewed. In this work various nodes are used with required computing resources situated in different geographic location.

Load balancing 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 while some others are under loaded.[2] Load balancing in the cloud computing environment has an important impact on the performance. 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 the game theory to the load balancing strategy to improve the efficiency in the public cloud environment. . When the environment is very large and complex, these divisions simplify the load balancing. The cloud has a main controller that chooses the suitable partitions for arriving jobs while the balancer for each cloud partition chooses the best load balancing strategy [4].

2. RELATED WORK

Load balancing is one of the central issues in cloud computing [5]. It is a mechanism that distributes the dynamic local workload evenly across all the nodes in the whole cloud to avoid a situation where some nodes are heavily loaded while others are idle or doing little work. It helps to achieve a high user satisfaction and resource utilization ratio, hence improving the overall performance and resource utility of the system. It also ensures that every computing resource is distributed efficiently and fairly [6]. It further prevents bottlenecks of the system which may occur due to load imbalance. When one or more components of any service fail, load balancing helps in continuation of the service by implementing fair-over, i.e. in provisioning and de-provisioning of instances of applications without fail.

There have been many studies of load balancing for the cloud environment. Load balancing [8] algorithm directly influences the effect of balancing the server workloads. Its main task is to decide how to choose the next server node and transfer a new connection request to it. Current main load balancing algorithm is divided into static algorithm and dynamic algorithm [9]. The static algorithm is easily carried into execution and takes less time, which doesn't refer to the states of the load nodes, but it can be only used in certain specific conditions. The common static algorithms are Round-Robin Scheduling Algorithm, Weighted Round-Robin Scheduling Algorithm, and Least-Connection Scheduling Algorithm etc. Of all, Round -Robin Scheduling Algorithm is the simplest one which could be most easily be carried out. However, it is only applicable to the circumstances in which all the nodes in cluster have the same processing ability. The dynamic algorithm like first come first serve is self-adaptive algorithm, which is better than static algorithm, and suitable for a great deal of requests which procreate different workloads, which would be unable to be forecasted [10]. Self-adaptive load balancing system mainly includes two processes: monitoring the load states of servers and assigning the request to the servers. The state supervision, which depends on the load information of each node in the cluster monitored and collected periodically by the front-end scheduler, raises the effect of load balance by monitoring load variety. At the same time, assigning the load carries on synchronous operation according to the load information from all nodes, that is, redistributing the load that needs to be done.

According to the analysis above, the ideal load balancing algorithm should achieve the following targets:

- Leave the collections, computing of load node information for each node; prevent the front-end scheduler from being system bottleneck.
- Reduce the complications of load balancing algorithm as far as possible.

3. SYSTEM MODEL AND PROBLEM FORMULATION

The paper describes the load balancing based on the cloud partitioning based on geographical locations is represented as follows:

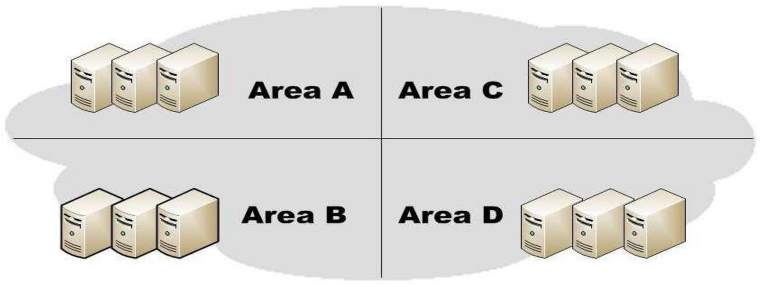


Fig 1. Cloud Partitioning

A public cloud consists of several nodes. All these nodes are partitioned into different partitions. Cloud partition is done to manage the public cloud. Each partition is representing the subarea of the public cloud. After creating the cloud partitions, the load balancing will be started.

The System model that we see in this paper for load balancing based on cloud partitioning consists of following elements (Fig 2):

- *Main Controller*, which handles the job (request) from the client by selecting the right partition.
- *Balancer*, that is responsible for handling partition.
- *Server (Node)*, that is responsible for the execution of job.

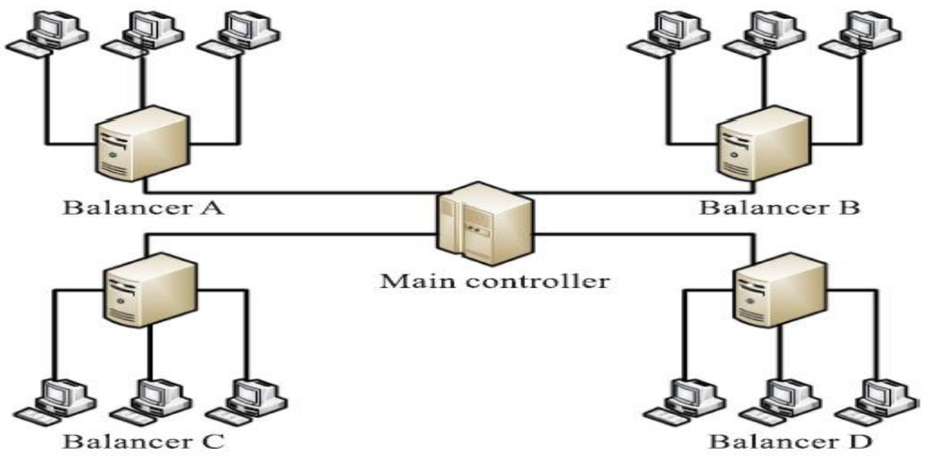


Fig 2. System model

Whenever a job arrives at the system from the client, the main controller checks to which partition the client belongs and then assigns the job to that partition, if it is in idle or normal status. If that partition is overloaded, the job is assigned based on the best partition strategy. The balancer of the partition accepts the job and assigns the job to server. This happens with the two cases: when the partition is at idle status, an improved round robin algorithm is used. When the partition is at normal status, Hungarian method of assignment problem is assigned.

A. Cloud Partition Status:

The Cloud partition status can be of three types:

- Idle: When the percentage of nodes with idle status exceeds α
- Normal: When the percentage of nodes with normal status exceeds β
- Overloaded: When the percentage of nodes with overloaded status exceeds γ

Note: α, β, γ are the parameters set by cloud partition balancers.

4. CLOUD PARTITION STRATEGY

Whenever, a job arrives at the main controller the main controller selects the partition for executing the job. The main controller selects the partition to which the job belongs, if the status of that partition is idle or normal. If the status is overloaded, a best partition searching algorithm is used to choose the partition for the execution of job. The best partition searching algorithm is as follows:

Algorithm 1

```
begin
  while job do
      searchBestPartition (job);
if partitionState == idle || partitionState == normal then
```

```
    Send Job to Partition;
else
    Search for another Partition;
end if
end while
end
```

Once the job is assigned to the partition, the job is executed based on two cases. If the partition is at idle status, the improved Round Robin algorithm is used. If the partition is at normal status, the Hungarian method of assignment problem is used.

A. Load balancing when partition is idle:

When the partition is at idle status, the resources will be available and relatively few jobs will be arriving. The Round Robin algorithm is one of the simplest load balancing algorithms. This is the algorithm in which the job is assigned based on the time slices. But the algorithm does not maintain any status information. Thus, the algorithm is improved that contains the status information and hence, the name improved Round Robin algorithm.

The algorithm is similar to Round Robin algorithm, but a status table is maintained at the partition that contains the list of all servers of that partition along with their load degree, and the list is sorted based on the load degree from lowest to the highest [12]. Whenever, there is a job, that job is assigned to the first server in the status table and the status table is refreshed.

B. Load balancing when partition is normal:

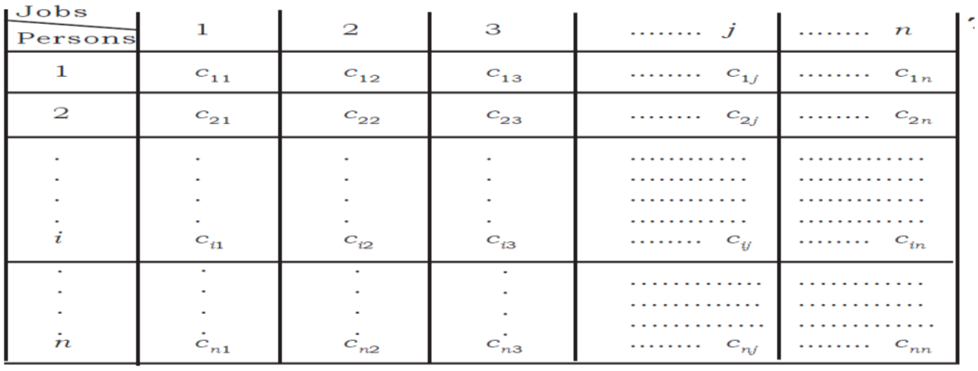
When the partition is at normal status, the jobs will be arriving faster than that of the idle condition. Thus, different algorithm is used for load balancing. The Hungarian method of assignment problem is used in that case.

Before applying the algorithm first, the assignment problem should be well understood [14]. If there are ‘n’ machines available and ‘n’ persons are engaged at different rates to operate them. Which operator should be assigned to which machine to ensure maximum efficiency? We have to find such an assignment by which the machine gets maximum efficiency on minimum duration. Such problems are known as "assignment problems".

Formulation of the problem:

Let there are n jobs and n persons are available with different skills. If the cost of doing j th work by i th person is c_{ij}. Then the cost matrix is given in the table below:

Table: 1. Table showing cost matrix



Jobs / Persons	1	2	3	.... j	.... n
1	c_{11}	c_{12}	c_{13}	.... c_{1j}	.... c_{1n}
2	c_{21}	c_{22}	c_{23}	.... c_{2j}	.... c_{2n}
⋮ i	⋮ c_{i1}	⋮ c_{i2}	⋮ c_{i3}	.... c_{ij}	.... c_{in}
⋮ n	⋮ c_{n1}	⋮ c_{n2}	⋮ c_{n3}	.... c_{nj}	.... c_{nn}

In order to find the proper assignment it is essential to know the Hungarian method. This method is dependent upon two vital theorems, stated as below.

Theorem 1: If a constant is added (or subtracted) to every element of any row (or column) of the cost matrix $[c_{ij}]$ in an assignment problem then an assignment which minimizes the total cost for the new matrix will also minimize the total cost matrix.

Theorem 2: If all $c_{ij} \geq 0$ and there exists a solution

$x_{ij} = X_{ij}$ such that $\sum c_{ij}\, x_{ij} = 0$.

then this solution is an optimal solution, i.e., minimizes z.

The computational procedure is given as under:

Step I (A) Row reduction:

Subtract the minimum entry of each row from all the entries of the respective row in the cost matrix.

(B) Column reduction:

After completion of row reduction, subtract the minimum entry of each column from all the entries of the respective column.

Step II Zero assignment:

(A) Starting with first row of the matrix received in first step, examine the rows one by one until a row containing exactly one zero is found. Then an experimental assignment indicated by '$\square$' is marked to that zero. Now cross all the zeros in the column in which the assignment is made. This procedure should be adopted for each row assignment.

(B) When the set of rows has been completely examined, an identical procedure is applied successively to columns. Starting with column 1, examine all columns until a column containing exactly one zero is found. Then make an experimental assignment in that position and cross other zeros in the row in which the assignment was made.

Continue these successive operations on rows and columns until all zero's have either been assigned or crossed-out.

Now there are two possibilities:

(a) Either all the zeros are assigned or crossed out, i.e., we get the maximal assignment.

or

(b) At least two zeros are remained by assignment or by crossing out in each row or column. In this situation we try to exclude some of the zeros by trial and error method.

This completes the second step. After this step we can get two situations.

(i) Total assigned zero's = n

The assignment is optimal.

(ii)Total assigned zero's < n

Use step III and onwards.

Step III: Draw of minimum lines to cover zero's

In order to cover all the zero's at least once you may adopt the following procedure.

(i) Marks (√) to all rows in which the assignment has not been done.

(ii) See the position of zero in marked (√) row and then mark (√) to the corresponding column.

(iii) See the marked (√) column and find the position of assigned zero's and then mark (√) to the corresponding rows which are not marked till now.

(iv) Repeat the procedure (ii) and (iii) till the completion of marking.

(v) Draw the lines through **unmarked rows** and **marked columns**.

Note: If the above method does not work then make an arbitrary assignment and then follow step IV.

Step IV: Select the smallest element from the uncovered elements.

(i) Subtract this smallest element from all those elements which are not covered.

(ii) Add this smallest element to all those elements which are at the intersection of two lines.

Step V: Thus we have increased the number of zero's. Now, modify the matrix with the help of step II and find the required assignment.

5. CONCLUSION AND FUTURE WORK

Load balancing in the cloud computing environment is the major issue and has major impact on the performance. Good load balancing of the cloud will provide a more efficient approach of accessing the cloud and improves user satisfaction.

The proposed work is an attempt to introduce 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. Even though the different strategies exist with this dynamic load balancing model, it does not inhibit the process of efficient achievement of cloud access and improvement of user satisfaction.

Since the proposed work is more into the conceptual framework, more improvisations have to be done in the future to resolve the issues.

➢ ***Cloud division rules*:** Cloud division is not a simple problem. The process has to be done with much of importance. Cloud division should be done in proper way. 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).

➢ ***Set the refresh period*:** 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 reasonable refresh periods.

REFERENCES

[1] http://searchcloudcomputing.techtarget.com/definition/cloud-computing.

[2] http://computer.howstuffworks.com/cloud-computing/cloud-computing.htm.

[3] http://www.arubacloud.com/cloud-computing/load-balancers.aspx.

[4] http://kemptechnologies.com/in/cloud-load-balancers/kemp-geo-loadmaster-and-hybrid-cloud-perfect-cloud-load-balancing-combination/.

[5] http://www.researchgate.net/profile/Gaochao_Xu/publication/260653020_A_load_balancing_model_based_on_cloud_partitioning_for_the_public_cloud/links/02e7e53aa69fed245c000000.pdf.

[6] http://www.ripublication.com/irph/ijict_spl/ijictv4n16spl_01.pdf.

[7] http://www.ijcsit.com/docs/Volume%205/vol5issue03/ijcsit20140503288.pdf.

[8] http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6449405.

[9] http://www.ijetae.com/files/Volume4Issue7/IJETAE_0714_117.pdf.

[10] http://www.iosrjournals.org/iosr-jce/papers/Vol16-issue1/Version-6/O016168287.pdf.

[11] http://www.ijarcce.com/upload/2013/december/IJARCCE6B-A-rajesh_A_Survey_on_Load_ Balancing%20%281%29.pdf

[12] http://www.ijstr.org/final-print/nov2013/A-Survey-Of-Various-Load-Balancing-Techniques-And-Challenges-In-Cloud-Computing.pdf.

[13] http://www.math.harvard.edu/archive/20_spring_05/handouts/assignment_overheads.pdf.

[14] https://faculty.sau.edu.sa/filedownload/ doc-6-pdf-a66b4d41a1b21574947d0d6c00e1de7a-original.pdf.