ISSN (Online): 2349-7084
GLOBAL IMPACT FACTOR 0.238
ISRA JIF 0.351

INTERNATIONAL JOURNAL OF COMPUTER ENGINEERING IN RESEARCH TRENDS
VOLUME 1, ISSUE 6, DECEMBER 2014, PP 367-372

Cloud Partitioning of Load Balancing Using Round Robin Model

[1] M.V.L.SOWJANYA, [2] D.RAVIKIRAN

[1] M.Tech Research Scholar, Priyadarshini Institute of Technology and Science for Women
[2] Professor, Priyadarshini Institute of Technology and Science for Women

Abstract: The purpose of load balancing is to look up the performance of a cloud environment through an appropriate circulation strategy. Good load balancing will construct cloud computing for more stability and efficiency. This paper introduces a better round robin model for the public cloud based on the cloud partitioning concept with a switch mechanism to choose different strategies for different situations. Load balancing is the process of giving out of workload among different nodes or processor. It will introduces a enhanced approach for public cloud load distribution using screening and game theory concept to increase the presentation of the system.

Key words: load balancing model; public cloud; cloud partition; game theory

1. INTRODUCTION

In the cloud technology users will introduce new types of services in that we have to change our life. Without paying an awareness of details we get new services [2]. NIST gave a definition of cloud computing as a model for enabling everywhere, expedient, on-demand network access to a collective pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be swiftly delivered and released with minimal management endeavor or service provider interaction [3]. More and more people pay interest to cloud computing [4, 5]. Cloud computing is systematic and scalable but preserving 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.

In a job arrival pattern also not expected and the capability of each node cloud be different, for load balancing problem, workload control is very important to improve system performance and sustain stability. Load balancing is the development of attractive performance of a parallel and distributed system via the redistribution of load among different processing units or nodes. Load balancing is corresponding to, "In the scenario of circulated network environment of computing hosts, the functioning of the system is heavily dependent upon dividing up work effectively across the several participating nodes".

A round robin is an arrangement of choosing all elements in a group equally in some rational order, usually from the top to the bottom of a list and then starting again at the top of the list and so on. A simple way to think of round robin is that it is about "taking turns". Thus, this model divides the public cloud into several cloud partitions. Load balancing simplification is based on environment is very large and complex. 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.

2. RELATED WORK

We get more studies of load balancing for the cloud environment. Alder was explained the Load balancing in cloud computing who began the tools and methods normally used for load them balancing in the cloud. In Chaczko et al [8] report arrives a new problem in load balancing in the cloud that needs new architectures to adapt to many modifications. In that they were explained about load balancing plays in improving the presentation and maintaining strength.

Here we are discussing about load balancing algorithms, such as Round Robin, Equally Spread Current Execution Algorithm, and Throttled Load

ISSN (Online): 2349-7084
GLOBAL IMPACT FACTOR 0.238
ISRA JIF 0.351

INTERNATIONAL JOURNAL OF COMPUTER ENGINEERING IN RESEARCH TRENDS
VOLUME 1, ISSUE 6, DECEMBER 2014, PP 367-372

Balancing.

Nishant et al [9] used the Throttled load optimization method in nodes load balancing. Randles et al.[10] checking the performance time and cost. He gave a compared analysis of some algorithms in cloud computing. In this scenario concluded that the ESCE algorithm and throttled algorithm are better than the Round Robin algorithm. Some of the classical loads balancing methods are similar to the allowance 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.

3. SYSTEM MODEL

Cloud architectures are mainly divided into three categories: public, private, and partner. The most familiar model of cloud computing to many consumers is the public cloud model, under which cloud services are provided in a virtualized environment, constructed using pooled shared physical resources, and accessible over a public network such as the internet. Public clouds, however, provide services to multiple clients using the same shared infrastructure. Public cloud has contains subarea that is cloud computing with partitions based on the geographic locations. The architecture is shown in Fig.1.

Cloud partitioning model follow the load balancing strategy. Load balancing will start after creation of cloud partitioning: The cloud has a main controller (MC) which decides the suitable partitions for new jobs.

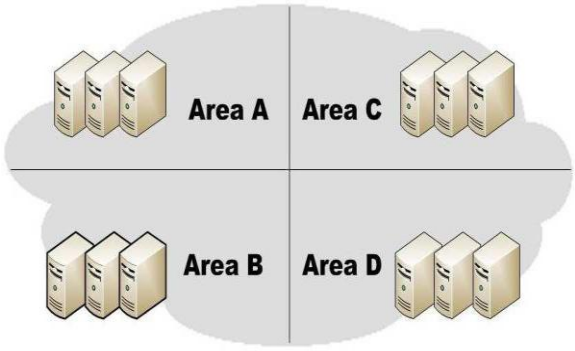


Fig.1 typical cloud partitioning.

Balancer has multiple servers attached to it. It keeps the record of all status information. Initially a request arrives at the system. The best partition search strategy helps to decide to which partition the request has to be assigned. The whole process is shown in Fig.2.

3.1 Main controller and balancers

The main controller and the balancers have done the load balancing solution. Main controller distributes the planned jobs to the suitable cloud partition based on execution time of the job. Main controller contains the configuration information of each partition. Based on the job length the main controller decides to which partition he job has to be assigned to get the response in less time. The balancers in each partition collect the status information from every node and then choose the right plan to distribute the jobs. In Fig.3 we can see a relationship between the balancers and the main controller.

3.2 Assigning jobs to the cloud partition

The first step is to choose the right partition after the job arrives at the public cloud. The cloud partition status can be divided into three types:

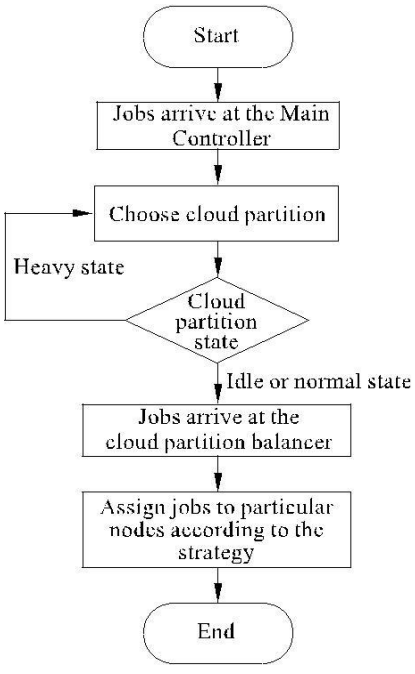


Fig. 2 Job assignment strategy.

(1) Overload: When the percentage of the overloaded nodes exceeds gamma, change to overloaded status.

ISSN (Online): 2349-7084
GLOBAL IMPACT FACTOR 0.238
ISRA JIF 0.351

INTERNATIONAL JOURNAL OF COMPUTER ENGINEERING IN RESEARCH TRENDS
VOLUME 1, ISSUE 6, DECEMBER 2014, PP 367-372

(2) Normal: When the percentage of the normal nodes exceeds beta, change to normal load status.
(3) Idle: When the percentage of idle nodes exceeds alpha, change to idle status.

The status information will refreshed by the main controller. It has to communicate with the balancers frequently. The main controller then transmits the jobs using the following approach:
The main controller inquiries the cloud partition where job is located when the job i appears at the system. If this location's status is idle or normal, the job is switched locally. If not, another cloud partition is found that is not overloaded. The algorithm is shown in Algorithm 1.

3.3 Assigning jobs to the nodes in the cloud partition

The cloud partition balancer collects load information from every node to evaluate the cloud partition status. This evaluation of each node's load status is very important. The first task is to define the load degree of each nodes.

Algorithm 1 Best Partition Searching

```
Begin
While job do
Search BestPartition
(job);
If partitionState == idle || partition State ==
normal then
    Send Job to Partition;
  Else
    Search for another
  Partition;
    End if
End
while
  End
```

The various static parameters and dynamic parameters are related to node load degree. The static parameters contain the number of CPU's, the CPU giving out speeds, the memory size, etc. Dynamic parameters are the memory use ratio, the CPU use ratio, the network bandwidth, etc. The load degree is calculated from these parameters as below:

Step 1 Define a load parameter set: Define a load parameter set: F = {F1; F2;…; Fm} with each Fi($1 \le i \le m$; Fi € [0, 1]) parameter being either static or dynamic.

Step 2 Compute the load degree as:

$$\text{Load degree (N)} = \sum_{i=1}^{m} \alpha i \, Fi$$

αi ($\sum_{i=1}^{n} \alpha i = 1$) are weights that may differ for different kinds of jobs. N represents the current node.

Step 3 Define estimate benchmarks. Calculate the average cloud partition degree from the node load degree statistics as:

$$Load\_degree_{avg} = \sum_{i=1}^{n} load\_degree(Ni) / n$$

The bench mark Load_degreehigh is then set for different situations based on the Load_degreeavg.

Step 4 Three nodes load status levels are then defined as:

Idle When
Load_degree (N) = 0;
In this node there is no job being processed so the status is charged to Idle.
Normal For
$0 < Load\_degree(N) \le Load\_degreehigh$,
the node is normal and it can process other jobs.
Overloaded When
Load_degreehigh ≤ Load degree (N);

The node does not exist and not obtain jobs until it returns to the normal.The cloud partition balancers are created a load status tables it takes an input from load degree results. Each every fixed period T we get balancer it has a Load Status Table and refreshes. If you want to calculate the partition status you can use balancer of tables. Each partition position has a different load balancing solution. When a job appears at a cloud partition, the balancer gives the job to the nodes based on its current load approach. This approach is changed by the balancers as the cloud partition position changes.

ISSN (Online): 2349-7084
GLOBAL IMPACT FACTOR 0.238
ISRA JIF 0.351

INTERNATIONAL JOURNAL OF COMPUTER ENGINEERING IN RESEARCH TRENDS
VOLUME 1, ISSUE 6, DECEMBER 2014, PP 367-372

4. CLOUD PARTITION LOAD BALANCING STRATEGY

4.1 Motivation

We can improve the performance of the entire cloud based on good load balance. However, there is no common technique that can adapt to all possible different conditions. Various techniques have been developed in civilizing existing solutions to resolution new problems.

Each particular technique has advantage in exacting area but not in all conditions. So, the current model combines several techniques and switches between the load balance techniques based on the system condition. A relatively simple technique can be used for the partition idle state with a more difficult technique for the normal state. The load balancers at that time switch methods as the status changes. Now, the idle status uses an enhanced Round Robin algorithm whereas the normal status applies a game theory based load balancing approach.

4.2 Load balance strategy for the idle status

Many computing sources are available and fairly few tasks are incoming while the cloud partition is idle. In this situation, this cloud partition has the capability to development tasks as fast as possible so a simple load balancing method can be used.

Here we can see many simple load balance algorithm techniques such as the Random algorithm, the Weight Round Robin, and the Dynamic Round Robin [12]. Here, the Round Robin algorithm maintains simplicity.

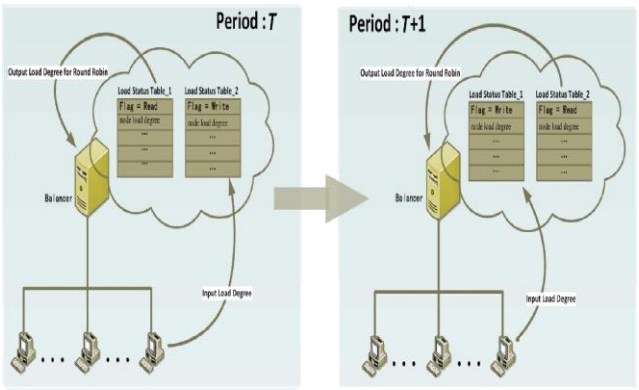
The Round Robin algorithm is one of the simplest load balancing algorithms, which exceeds each new demand to the next server in the queue. The algorithm does not record the condition of every connection thus it has no condition information. In the normal Round Robin algorithm, every node has an equal chance to be special. However, in a public cloud, the design and the presentation of each node will be not the same; thus, this method may overload some nodes. Thus, an improved Round Robin algorithm is old, which called "Round Robin based on the load degree evaluation".

The algorithm is still literally simple. Before the Round Robin step, the nodes in the load balancing table are planned based on the load degree from the lowest to the highest. The system makes a circular queue and walks throughout the queue again and again. Jobs will then be allotted to nodes with low load degrees. The node order will be altered when the balancer recharges the Load condition Table. Still, there may be examine and mark difference at the refresh period T. When the balance table is revived, at this immediate, if a job appears at the cloud partition, it will bring the contradictory problem. The system situation will have changed but the information will still used. This may lead to an incorrect load approach option and an incorrect nodes order. To determine this problem, two Load category Tables should be created as: Load Status Table_1 and Load Status Table_2. A flag is also allocated to each table to specify examine or mark.

When the flag = "Read", then the Round Robin based on the load degree evaluation algorithm is using this table.

When the flag = "Write", the table is mortal refreshed, new information is written into this table.

Thus, at each instant, one table offers the correct node locations in the queue for the enhanced Round Robin algorithm, while the other is mortal arranged with the efficient information. Once the data is recharged, the table flag is changed to "Read" and the other table's flag is changed to "Write". The two tables then alternate to solve the variation. The



Process is shown in Fig.3.

4.3 Load balancing strategy for the normal status

The jobs are arriving much faster than in the idle state and the position is far more difficult, so a different

ISSN (Online): 2349-7084
GLOBAL IMPACT FACTOR 0.238
ISRA JIF 0.351
INTERNATIONAL JOURNAL OF COMPUTER ENGINEERING IN RESEARCH TRENDS
VOLUME 1, ISSUE 6, DECEMBER 2014, PP 367-372

approach is used for the load balancing while the cloud partition is normal. Each user desires his tasks finished in the shortest time, so the public cloud requires methods that can entire jobs of all users with logical response time.

Penmatsa and chronopoulos planned a static load balancing approach based on game theory for distributed systems. And this work provides us with a new analysis of the load balance problem in the cloud atmosphere. As an implementation of distributed system, the load balancing in the cloud computing atmosphere can be viewed as a game.

Game theory has non-supportive games and supportive games. In cooperative games, the result makers ultimately come to an contract which is called a binding agreement. Each decision maker decides by comparing notes with each others. In non-cooperative games, each decision maker makes results only for his own benefit. The system then makes the Nash equilibrium, where each result maker makes the optimized decision. The Nash equilibrium is when each player in the game has selected approach and no player can advantage by changing his or her approach while the other player's approaches stay unchanged.

There have been many studies in using game theory for the load balancing. Grosu et al.[14] designed a load balancing approach based on game theory for the distributed systems as a non-supportive game using the distributed structure. They balanced this algorithm with other conventional methods to show that their algorithm was less difficulty with better presentation. Aote and Kharat [15] gave a dynamic load balancing model based on game theory. This model is linked on the dynamic load position of the system with the users being the choice makers in a non-supportive game.

Since the grid computing and cloud computing atmospheres are also distributed system, these algorithms can also be used in grid computing and cloud computing atmospheres. Earlier studies have exposed that the load balancing approach for a cloud partition in the normal load condition can be viewed as a non-supportive game, as explained here.

In this model, the most important step is finding the suitable value of sji. The existing model uses the method of Grosu et al.[14] called "the best reply" to calculate sji of each node, with a greedy algorithm then used to analyze sji for all nodes. This method gives the Nash equilibrium to minimize the response time of each job. The plan then changes as the node's position modify.

5. FUTURE WORKS

While this work is now a conceptual framework, additional work is desired to apply the framework and determine new problems. Some significant points are:

(1) **Cloud division rules:** Cloud division is not a easy problem. Thus, the framework will require a comprehensive cloud division methodology. For example, nodes in a group may be extreme from other nodes or there will be some groups in the same geographic area that are still far apart. The division rule should only base on the geographic location (region or state).

(2) **Find other load balance strategy:** Other load balance approaches may provide better results, so tests are required to balance different approaches. Many tests are required to pledge system availability and efficiency.

(3) **A better load status evaluation:** A good algorithm is desired to set Load_degree high and Load_degree low, and the estimate device requires being more widespread.

(4) **How to locate the refresh period:** In the data statistics analysis, the main controller and the cloud partition balancers require to restore the information at a fixed period. If the period is too short, the high frequency will manipulate the system presentation. If the period is too long, the information will be too previous to make good conclusion. Thus, tests and statistical tools are desired to set logical refresh periods.

6. CONCLUSSION

This paper introduces a better round robin model for the public cloud based on the cloud partitioning concept with a switch mechanism to choose different strategies for different situations. Load balancing is the process of giving out of workload among different nodes or processor. It will introduces a enhanced approach for public cloud load distribution using screening and game theory concept to increase the presentation of the system. Load balancing will construct cloud computing for more stability and efficiency

REFERENCES

[1] R. Hunter, The why of cloud, http://www.gartner.com/DisplayDocument?doc cd=226469&ref= gnoreg, 2012.

[2] M. D. Dikaiakos, D. Katsaros, P. Mehra, G. Pallis, and A. Vakali, Cloud computing: Distributed internet computing for IT and scientific research, Internet Computing, vol.13, no.5, pp.10-13, Sept.-Oct. 2009.

[3] P. Mell and T. Grance, The NIST definition of cloud computing, http://csrc.nist.gov/ publications/nistpubs/800-145/SP800-145.pdf, 2012.

[4] Microsoft Academic Research, Cloud computing, http:// libra.msra.cn/Keyword/6051/cloud-computing? query = cloud%20computing, 2012.

[5] Google Trends, Cloud computing, http://www.google.com/trends/explore#q=cloud %20computing, 2012.

[6] N. G. Shivaratri, P. Krueger, and M. Singhal, Load distributing for locally distributed systems, Computer, vol. 25, no. 12, pp. 33-44, Dec. 1992.

[7] B. Adler, Load balancing in the cloud: Tools, tips and techniques, http://www.rightscale. com/info center/white-papers/Load-Balancing-in-the-Cloud.pdf, 2012

[8] Z. Chaczko, V. Mahadevan, S. Aslanzadeh, and C. Mcdermid, Availability and load balancing in cloud computing, presented at the 2011 International Conference on Computer and Software Modeling, Singapore, 2011.

[9] K. Nishant, P. Sharma, V. Krishna, C. Gupta, K. P. Singh, International Conference on Computer Modeling and Simulation (UKSim), Cambridge shire, United Kingdom, Mar. 2012, pp. 28-30.

[10] 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, pp. 551-556.

[11] A. Rouse, Public cloud, http://searchcloudcomputing.techtarget.com/defi nition/public-cloud, 2012.

[12] D. MacVittie, Intro to load balancing for developers — the algorithms, https://devcentral.f5.com/blogs/us/intro-to-load-balancing-for-developers-ndash-the-algorithms, 2012.

[13] S. Penmatsa and A. T. Chronopoulos, Game-theoretic static balancing for distributed systems, load journal of Distributed computing pp.502-11-42.

[14] 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.

[15] S. Aote and M. U. Kharat, A game-theoretic model for dynamic load balancing in distributed systems, in Proc. the International Conference on Advances in Computing, Communication and Control (ICAC3 '09), New York, USA, 2009, pp. 235-238.