ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

Efficient Cloud Computing Load Balancing Using Cloud Partitioning and Game Theory in Public Cloud

P.Rahul[1], Dr.A.Senthil Kumar[2], Boney Cherian[3]

P.G. Scholar, Department of CSE, R.V.S. College of Engineering and Technology, Coimbatore, India[1].

Professor, Department of CSE, R.V.S. College of Engineering and Technology, Coimbatore, India[2].

P.G. Scholar, Department of CSE, R.V.S. College of Engineering and Technology, Coimbatore, India[3].

ABSTRACT: Cloud computing could be an idea that has several computers interconnected through a true time network like web. Cloud computing is economical and ascendable however to keep up the soundness of process many roles within the cloud computing could be a terribly tough drawback. The work arrival pattern cannot be foretold and also the capacities of every node within the cloud disagree. Thus for levelling the usage of web and connected resources has enhanced wide. As a result of this there's tremendous increase in employment. Therefore there's uneven distribution of this employment which ends in server overloading and should crash. In such the load, it's crucial to manage workloads to boost system performance and maintain stability. The load on each cloud is variable and addicted to numerous factors. Smart load levelling makes cloud computing additional economical and conjointly improves user satisfaction. The load levelling model given during this work is aimed toward the general public cloud that has various nodes with distributed computing resources in many alternative geographic locations. Thus, this model divides the general public cloud into many cloud partitions. Once the atmosphere is extremely massive and complicated, these divisions alter the load levelling. The cloud encompasses a main controller that chooses the acceptable partitions for incoming jobs whereas the balancer for every cloud partition chooses the most effective load levelling strategy.

KEYWORDS: Game Theory, Cloud partition, Load balancing

I. INTRODUCTION

The load equalization technique accustomed make certain that none of the node is in idle state whereas different nodes are being utilized". So as to balance the load among multiple nodes you\'ll be able to distribute the load to a different node that has gently loaded. so distributing the load throughout runtime is understood as Dynamic Load equalization technique. Load equalization algorithmic rule is divided into 2 classes as 1) Static and 2) Dynamic. In static load equalization algorithmic rule, all the knowledge concerning the system is understood prior to, and also the load equalization strategy has been created by load equalization algorithmic rule at compile time. These load equalization strategies are going to be unbroken perpetually throughout runtime of the system.

In distinction, dynamic algorithmic program is enforced at period, and therefore the load leveling ways amendment according to the statement of the system. Though, the dynamic algorithmic program has higher ability, it is sensitive to the accuracy of the load data or statement of system. Several researchers have projected many algorithms for load leveling. In cloud computing once a computation is requested by any system it is distributed to any or all the slaves existing in this cloud. Therefore the manner within which the distribution is being done should get the response from all the slaves at identical time so there must not be any expecting any specific computing machine system to reply before more processing might happen. However within the real time clouds heterogeneous computing devices exists and any process is execution time on the slave is needed to be calculable. Therefore the main feature that is should in any load balancer is that the uneven load distribution.

Central to those problems lies the institution of a good load equalization algorithmic rule. The load will be C.P.U. load, memory capability, delay or network load. Load equalization is that the method of distributing the load

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

among varied nodes of a distributed system to boost each resource utilization and job reaction time whereas conjointly avoiding a state of affairs wherever a number of the nodes area unit heavily loaded whereas alternative nodes area unit idle or doing little or no work. Load equalization ensures the processor within the system or every node within the network will close to the equal quantity of labor at any instant of your time. This system will be sender initiated, receiver initiated or symmetrical kind.

Load equalization in cloud computing systems is absolutely a challenge currently. Forever a distributed answer is needed. As a result of it's not forever much possible or value economical to take care of one or a lot of idle services even as to meet the desired demands. Jobs can't be appointed to applicable servers and purchasers singly for economical load equalization as cloud could be a terribly advanced structure and parts area unit gift throughout a good unfold space. Here some uncertainty is connected whereas jobs area unit appointed.

The load reconciliation model given during this work is geared toward the general public cloud that has varied nodes with distributed computing resources in many alternative geographic locations. Thus, this model divides the general public cloud into many cloud partitions. Once the surroundings are incredibly giant and sophisticated, these divisions change the load reconciliation. The cloud contains a main controller that chooses the appropriate partitions for inbound jobs whereas the balancer for every cloud partition chooses the simplest load reconciliation strategy.

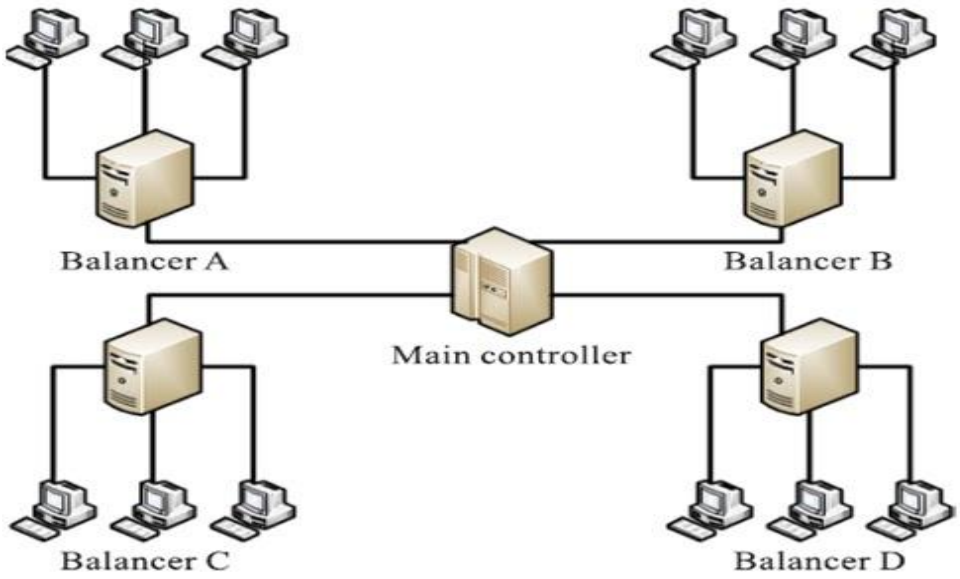


Fig.1 cloud partitioning in public cloud

II. RELATED WORK

Global programing is more classified into static and dynamic programing classes. In static programing processes area unit allotted to processors before the executions starts. On the opposite hand dynamic programing will assign the processes to the processors throughout the execution. Load sharing and cargo equalization area unit the more classifications of dynamic programing. Load sharing struggle to avoid the separate state in processors that stay idle whereas tasks contend for service at another processor. Load equalization additionally does identical however it goes one step before load sharing by trying to equalize the masses in the slightest degree processors. Load equalization is to make sure that each processor within the system will some identical quantity of labor at any purpose of your time.

Within the spherical robin processes area unit divided equally between all processors. Every new method is allotted to new processor in spherical robin order. The method allocation order is maintained on every processor domestically freelance of allocations from remote processors. With equal work spherical robin algorithmic rule is predicted to figure well. The minimally loaded method or reckoning on the load is chosen once process is formed.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

Load manager selects hosts for brand spanking new processes in order that the processor load confirms to same level the maximum amount as potential. From then to be had data on the system load state central load manager makes the load equalization judgment. This data is updated by remote processors that send a message whenever the load on them changes. In keeping with this algorithmic rule, the processes area unit allotted right away upon creation to hosts. Hosts for brand spanking new processes area unit chosen domestically while not causing remote messages. Every processor keeps a non-public copy of the system's load.

III.SYSTEM MODEL

There are several units of cloud computing categories with this work targeted on a public cloud. Cloud partitioning is used to management this massive cloud. A cloud partition is a subarea of the general cloud with divisions supported the geographic locations.

The load reconciliation strategy relies on the cloud partitioning conception. When making the cloud partitions, the load reconciliation then starts: once employment arrives. The system, with the most controllers deciding that cloud partition ought to receive the duty. The partition load balancer then decides a way to assign the roles to the nodes. Once the load standing of a cloud partition is normal, this partitioning may be accomplished regionally. If the cloud partition load standing isn't traditional, this job should be transferred to a different partition.

1.1 Main controller and balancers

The load balance resolution is finished by the most controller and the balancers. The main controller initial assigns jobs to the suitable cloud partition and so communicates with the balancers in every partition to refresh this standing information. Since the most controller deals with information for every partition, smaller information sets can lead to the upper process rates. The balancers in each partition gather the standing information from every node then pick the proper strategy to distribute the jobs. The affiliation between the balancers and conjointly the main controller is shown in Fig.1.

1.2 Assigning jobs to the cloud partition

The main controller 1st assigns jobs to the acceptable cloud partition so communicates with the balancers in every partition to refresh this standing data. Since the most controller deals with data for every partition, smaller knowledge sets can cause the upper process rates. The balancers in every partition gather the standing data from each node so opt for the correct strategy to distribute the roles. When a job arrives at the public cloud, the first step is to choose the right partition. The cloud partition status can be divided into three types:

(1) Idle: once the share of idle nodes exceeds α, modification to idle standing.

(2) Normal: once the share of the conventional nodes exceeds β, modification to traditional load standing.

(3) Overload: once the share of the overladen nodes exceeds γ, modification to overladen standing.

The parameters α, β and γ area unit set by the cloud partition balancers.

The main controller has got to communicate with the balancers often to refresh the standing info. The most controllers then dispatch the roles mistreatment the subsequent strategy: once job i arrives at the system, the most controller queries the cloud partition wherever job is found. If this location's standing is idle or traditional, the work is handled regionally. If not, another cloud partition is found that is not full.

Algorithm 1 Best Partition Searching

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

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

Send Job to Partition;
else
search for another Partition;
end if
end while
end

1.3 Assigning jobs to the nodes in the cloud partition

The cloud partition balancer gathers load information from every node to gauge the cloud partition standing. This analysis of each node's load standing is implausibly important. The first task is to stipulate the load degree of each node. The node load degree is expounded to varied static parameters and dynamic parameters. The static parameters embody the quantity of CPU's, the central C.P.U. process speeds, the memory size, etc. Dynamic parameters unit the memory utilization magnitude relation, the pc hardware utilization magnitude relation, the network metric, etc.

Step 1 Define a load parameter set: F= {F_1.F_2,....., F_m} with each $F_i(1 \leq i \leq m, F_i \in [0,1])$ parameter being either static or dynamic. m represents the total number of the parameters.

Step 2 Compute the load degree as:

Load_degree (N) = $\sum_{i=1}^{m} \propto_i F_i$

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

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

$$Load\_degree_{avg} = \frac{\sum_{i=1}^{n} Load\_degree(N_i)}{n}$$

The bench mark $Load\_degree_{high}$ is then set for different situations based on the $Load\_degree_{avg}$.

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

- **Idle** When Load_degree(N)=0;

  There is no job being processed by this node so the status is charged to Idle.

- **Normal** For 0 < Load_degree(N)$\leq$ $Load\_degree_{high}$, the node is normal and it can process other jobs.

- **Overloaded** When $Load\_degree_{high} \leq$ Load_degree(N), the node is not available and cannot receive jobs until it returns to the normal.

1.4 Load balance strategy for all nodes

- Load balance strategy for the idle status v

- Load balancing strategy for the normal status

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

When the cloud partition is normal, jobs are arriving much faster than in the idle state and the situation is far more complex, so a different strategy is used for the load balancing. Each user wants his jobs completed in the shortest time, so the public cloud needs a method that can complete the jobs of all users with reasonable response time.

Game theory has non-cooperative games and cooperative games. In cooperative games, the decision makers eventually come to an agreement which is called a binding agreement. Each decision maker decides by comparing notes with each other's. In non-cooperative games, each decision maker makes decisions only for his own benefit. The system then reaches the Nash equilibrium, where each decision maker makes the optimized decision. The Nash equilibrium is when each player in the game has chosen a strategy and no player can benefit by changing his or her strategy while the other player's strategies remain unchanged.

The players in the game are the nodes and the jobs. Suppose there are n nodes in the current cloud partition with N jobs arriving, and then define the following parameters:

μ_i: Processing ability of each node, i=1,....,n

ϕ_j: Time spending of each job

$\Phi = \sum_{j=1}^{N} \phi_j$: Time spent by the entire cloud partition $\Phi < \sum_{i=1}^{N} \mu_i$

S_{ji}: Fraction of job j that assigned to node i $\sum_{i=1}^{N} S_{ji} = 1 \; and \; 0 \leq S_{ji} \leq 1$

In this model, the most important step is finding the appropriate value of S_{ji}. The current model uses the method called "the best reply" to calculate S_{ji} of each node, with a greedy algorithm then used to calculate S_{ji} for all nodes. This procedure gives the Nash equilibrium to minimize the response time of each job. The strategy then changes as the node's statuses change.

3.5 Performance Comparison

In this module we compare the results of existing and proposed system with following parameters as load, response time and throughput.

IV. EXPERIMENT RESULT

Before the process starts have to create cloud environment. This below figure simply represents the partition of the cloud into different zones.

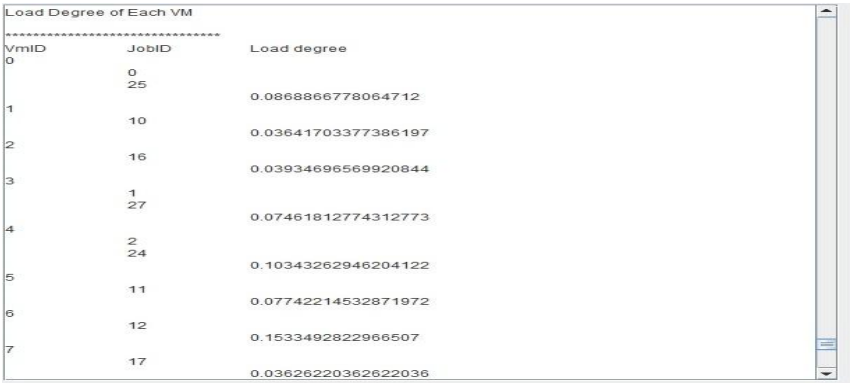
ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

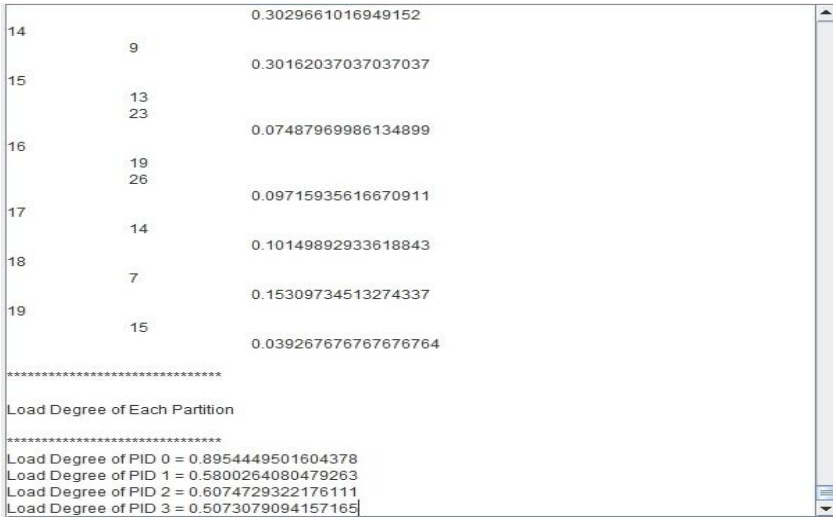
(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

After that calculating the load degree of the virtual machine and allocating jobs into the virtual machine.



To checking the load degree of the virtual machine and the same process continue for upcoming jobs.



V. CONCLUSION AND FUTURE WORK

The latent period and knowledge transfer value could be a challenge of each engineer to develop the merchandise which will increase the business performance and high client satisfaction within the cloud primarily based sector. The many methods lack economic planning and cargo leveling resource allocation techniques resulting in increased operational value and provides less client satisfaction. Load leveling within the cloud computing setting has a very important impact on the performance. Sensible load leveling makes cloud computing a lot of economical and improves user satisfaction. During this paper we have projected a much better load balance model for the duty Seeker's net Portal supported the cloud partitioning thought with a switch mechanism to settle on completely different methods for various things. Thus, this model divides the general public cloud into many cloud partitions. Once the setting is extremely giant and complicated, these divisions alter the load leveling. The cloud contains a main controller that chooses the acceptable partitions for inward jobs supported arrival date. Therefore with cloud partitioning thought it is doable to produce sensible load leveling and thus raising the general performance of cloud setting and user satisfaction.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

REFERENCES

[1]. Gaochao Xu, Junjie Pang, and Xiaodong Fu [2013], "A Load Balancing Model Based on Cloud Partitioning for the Public Cloud", IEEE transactions on cloud computing.
[2]. M.D. Dikaiakos, D. Katsaros, P. Mehra, G. Pallis, & A.Vakali [2009], "Cloud computing: Distributed internet computing for IT and scientific research", Internet Computing, vol.13, no.5, pp.10-13.
[3]. M. Randles, D. Lamb, and A. Taleb-Bendiab [2010], "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,pp. 551-556.
[4]. S.Aote and M. U.Kharat [2009], "A game-theoretic model for dynamic load balancing in distributed systems, in proc". The international conference on advances in computing, Communication and control.
[5]. D. Grosu, A. T. Chronopoulos, and M. Y. Leung [2009], "Load balancing in distributed systems: An approach using cooperative games, in Proc". 16th IEEE Intl. Parallel and Distributed Processing Symp.
[6]. S. Penmatsa and A. T. Chronopoulos [2011], "Game-theoretic static load balancing for distributed systems, Journal of Parallel and Distributed Computing" vol. 71, no. 4.
[7]. K. Nishant, P. Sharma, V. Krishna, C. Gupta, K. P. Singh, N. Nitin, and R. Rastogi [2012], "Load balancing of nodes in cloud using ant colony optimization, in Proc". 14th International Conference on Computer Modelling and Simulation (UKSim), Cambridgeshire, United Kingdom, pp. 28-30.
[8]. D. MacVittie [2012], Intro to load balancing for developers, the algorithms, https://devcentral.f5.com/blogs/us/introto-load-balancing-for-developers-ndash-the-algorithms.

BIOGRAPHY

P.Rahul is pursuing Master of Engineering in Computer Science and Engineering in RVS. College of Engineering and Technology, Coimbatore, India.

Dr.A.Senthil Kumar is Professor in Department of Computer Science, RVS. College of Engineering and Technology,Coimbatore,India.

Boney Cherian is pursuing Master of Engineering in Computer Science Engineering in RVS. College of Engineering and Technology, Coimbatore, India.