IJERMT

International Journal Of Engineering Research & Management Technology

ISSN: 2348-4039

Email: editor@ijermt.org March- 2014 Volume-1, Issue-2 Website: ijermt.org

PRIORITY BASED ENHANCED HTV DYNAMIC LOAD BALANCING ALGORITHM IN CLOUD COMPUTING

Srishti Agarwal,
Research Scholar
SRM University, NCR Campus,
Ghaziabad,

Nipun Kumar
Research Scholar
Jaypee Inst. Information technology
Noida (U.P.)

Abstract-

Cloud computing is a very rapidly growing technology now a days. Almost every organization is taking advantage of Cloud computing. Cloud computing is an Internet –based development in which virtual resources are provided as a service over the internet .The more the number of users on the cloud ,the greater will be the load. Cloud service providers are required to balance this load effectively and efficiently so as not to degrade the performance. So load balancing is a very important issue which aims at distributing the load across multiple machines in an even and fair manner so that no single node is overwhelmed. In this paper, we aim to manage the distribution of resources in such a manner so as to avoid system bottlenecks. This paper will be presenting the issues of existing load balancing algorithms and also presenting the algorithm to overcome those issues.

Keywords: Cloud computing Load balancing, Virtual machine, Resource allocation.

I. INTRODUCTION

Cloud computing is a kind of distributed computing where different massively scalable IT related resources or capabilities are provided to a number of external users as a service using internet. Now a days, Cloud computing is widely used in almost every organization , the reason being the beautiful way in which it provides the usage of its virtual resources and scalability. The speed with which the popularity of the cloud computing is growing , it is very certain to predict that in the near future Cloud computing will have a very significant impact on almost every area whether it is business , education , science and engineering etc.

The main feature of the Cloud computing is that it makes all the resources available at one place in the form of a cluster and the resources are allocated to the users according to their requests. This cluster based approach helps in achieving the maximum CPU utilization and reduces the efforts of users to access the cloud resources.

[NIST (The National Institute of Standards and Technology) Def] : Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction[6]. This cloud model promotes availability and is composed of five essential **characteristics,** three **service models**, and four **deployment models.**

According to NIST the five characteristics are: On-demand self-service, Broad network access , Resource pooling(Location independence) , Rapid elasticity , Measured service.

The three service models are:
1. Software as a service: Users can use the provider's applications using some web browser.
2. Platform as a service: Client can deploy applications on to cloud using some programming language and tool.
3. Infrastructure as a service: Client can deploy applications and can run some arbitrary software.

Despite of the functionalities of cloud , cloud also have some issues (virtual machine migration , cost , security , load balancing , power management) , out of which load balancing is the most hot topic . As more and more industries are migrating towards Cloud computing, the no. of users on cloud will be increasing which leads to issue of load balancing on cloud that needs to be handled efficiently. Load imbalance is a condition which occurs when some nodes have many requests to service while others are idle means there is no proper utilization of available resources.

II. LOAD BALANCING

The reliability of clouds depends on the way it handles the load, so to overcome the issue of load ,clouds must be featured with an efficient load balancing mechanism.[3] Load balancing is a mechanism which ensures that all the dynamic workload is distributed among all the multiple nodes in an evenly and efficient manner in such a way so as to avoid a condition where some nodes are heavily loaded while leaving some nodes to be idle or lightly loaded.
Load balancing is the pre-requirement for increasing the cloud performance and completely utilizing the resources. It also ensures that every computing resource is distributed efficiently and evenly. Load balancing helps in reducing the bandwidth usage which results in decreasing the cost of machine and maximizing the services. Load balancing aims to optimize the resource utilization, maximize the throughput, minimize the response time, and avoid overheads of any one of resource.

III. EXISTING LOAD BALANCING ALGORITHMS

Some of the existing load balancing algorithms has been discussed here:

- **EQUALLY SPREAD CURRENT EXECUTION ALGORITHM:** As the name suggests, this algorithm works by equally spreading the load on different virtual machines.[3] This algorithm requires a load balancer which queue up all the jobs that are asked for execution and then hand over them to different virtual machines. It distributes the load randomly by checking the size and transfer the load to that virtual machine which is lightly loaded or that can handle the task easily, take less time, and give maximum throughput.

- **THROTTLED LOAD BALANCING ALGORITHM:** This algorithm starts working by finding the most suitable virtual machine for assigning a given particular job. Throttled load balancer maintains the record of each virtual machine [3] .The client first asks the throttled load balancer to look for the suitable virtual machine according to the need of job and then the load balancer allocates the ideal virtual machine to the incoming job. If no virtual machine is available then the client request will be queued for fast processing.

- **ROUND ROBIN ALGORITHM:** Round robin uses a time slicing mechanism. As the name itself says that every node is given a particular time slot (time slice) in which they have to perform their task. So every node has to wait for their turn or time slot to perform their task. The tasks are divided and allocated to all nodes in a round robin fashion. Although the load distributions between all nodes are equal but the processing time of each node is different. So it may happen that at some point of time some nodes may be heavily loaded while others remain idle. This is the drawback of this algorithm.

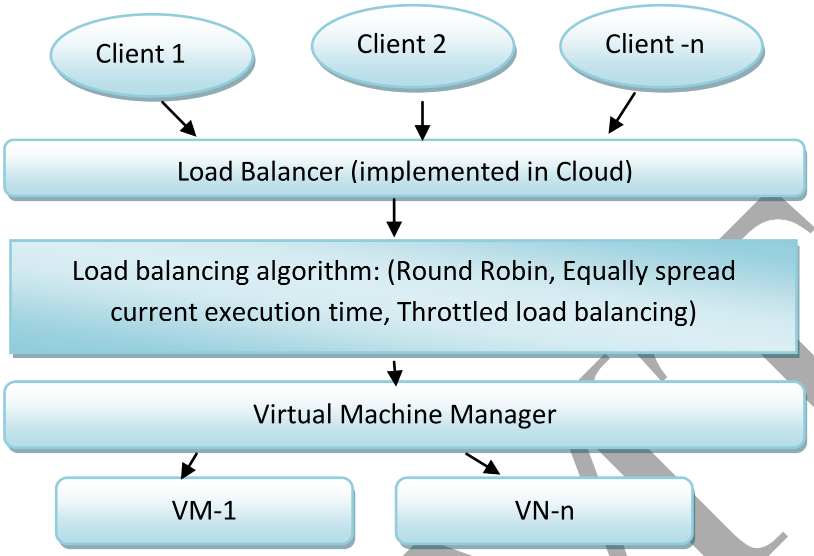
Client 1
Client 2
Client -n
Load Balancer (implemented in Cloud)
Load balancing algorithm: (Round Robin, Equally spread current execution time, Throttled load balancing)
Virtual Machine Manager
VM-1
VN-n

FIG 1. LOAD BALANCING

IV. PROBLEM DESCRIPTION

Although there are many algorithms available for load balancing in Cloud computing but on analyzing them thoroughly we still feel the need of further research so as to improve the performance and efficiency of the algorithms.

The current load balancing algorithm [1] distributes the workload among all the nodes in a round robin fashion i.e. it allocates the first request to the very first node in the queue, then the second request is allocated to the second node in the queue if enough resources are available on that node otherwise it moves to the next node in the queue, and when the end of the queue is reached it will again start from the very first node in the queue. Here the problem is that there is no resource monitoring hence it does not have any idea about the node whom it assigns the request is heavily loaded or lightly loaded. So some nodes may be heavily loaded while others are idle but still round robin will assign the request to that heavily loaded node if it is the next node in the queue which will degrade the performance and efficiency of that heavily loaded node.

So to overcome this problem a new algorithm (HTV Dynamic Load Balancing Algorithm) [1] was proposed in which continuous monitoring of the resources are done to know the status of each and every node and queue is maintained in which the weight factor will be stored and update whenever continuous monitoring is done. When request comes, the resources will be allocated from the information present in the queue dynamically to balance the load on nodes.

But still there is some problem which needs to be worked out. The problem of priority of the requests (tasks) that needs to be executed. Some tasks have higher priority that the other ones. Such task needs to be serviced prior than the other ones.

V. PROPOSED ALGORITHM

We are proposing an algorithm in which there will be a continuous monitoring of the resources so as to know the status of available resources on each node as well as there will also be a concept of priority of the incoming tasks or requests that needs to be serviced.

PSEUDO CODE

```
Step 1: [Calculate Load Factor X];
    X ← (Total _Resources - Used _Resources);

    // where X is free memory in terms of percentage.

Step 2: [Calculate Performance Factor Y];

    Y1 ← average (current_response_time)

    Y  ←  Y1 – (previously calculated Y1)

    Y ←  Y/(previous Y1) * 100  // counting Y in terms of previously counted Y1.

Step 3: [Finding Z];

    Z ← X – Y:

    If ( Z< 0 )

    Z = 0;

Step 4: [find minimum of all Z expect the nodes with Z value 0]

    Min_Z= min (all Z's)

Step 5: [Find Min_factor and divide all Z by that factor]

     Min_factor ← Min_Z

     Z ← Z/Min_factor

Step 6: [Generate Dynamic Queue on base of  Z ]

Step 7:  [Arrange the incoming tasks in ascending order of their priority]

Step 8: [Classify each task in their priority queue]

Step 9: Repeat until all tasks are allocated

          Or until all servers (virtual machines) are fully loaded;

Step 10:  dequeue task from the queue and allocate to virtual machine using priority based load
balancing algorithm in the ratio 3:2:1
```

EXPLANATION

Step-1: The value of load factor X is calculated by using the fomula: X = (Total_Resources – Used_Resources) , by this we get the available free load on nodes.

Step-2: To get an idea about the increase or decrease of performance , the performance factor Y is calculated by sending a request at regular interval to all nodes and the response time is calculated(request time +response time). Every time the value of Y will be different and averaging them Y1 will be calculated , and then the previously calculated Y1 will be subtracted from current values Y1 to get the performance .

Step-3: Calculate Z (mathematical function to count parameter value for each node) using formula : Z = X-Y , here we are interested in node with lowest response time hence we subtract Y from X.

Step-4: Once Z is calculated then the minimum of all Z which we calculated is stored in Min_Z.

Step-5: After getting the Min_factor , divide all the Z values by that factor .

Step-6: On the basis of this Z a queue is generated.

Step-7: The incoming tasks will be arranged on the basis of their priority.

Step-8: All the tasks will be classified in their respective priority queues.

Step-9: Loop will continue until all the tasks are allocated or all the machines will be fully loaded.

Step-10: The tasks will be allocated to the virtual machines and simultaneously will be removed from the queue.

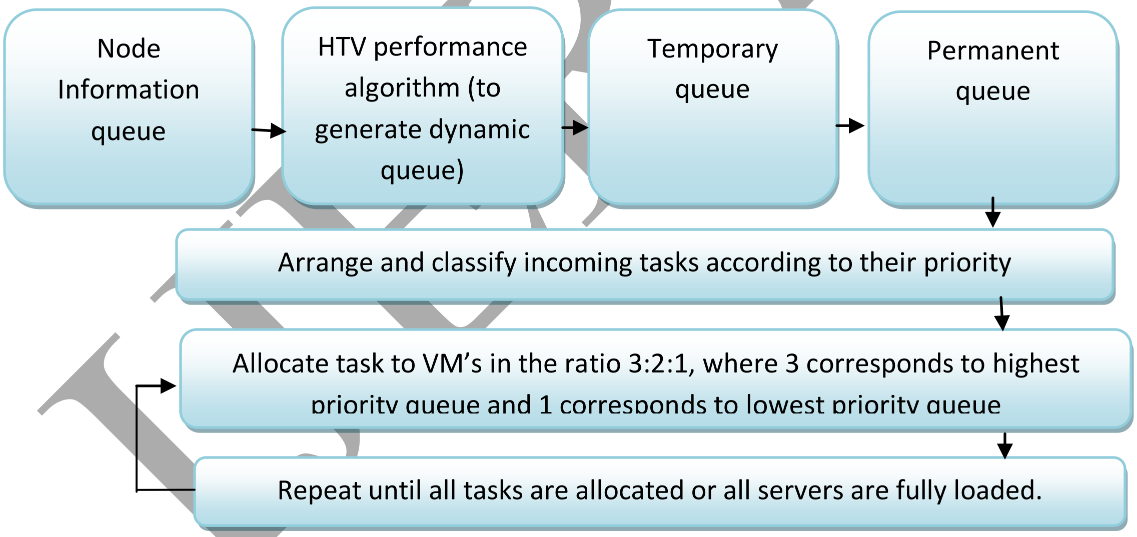
Node Information queue
HTV performance algorithm (to generate dynamic queue)
Temporary queue
Permanent queue
Arrange and classify incoming tasks according to their priority
Allocate task to VM's in the ratio 3:2:1, where 3 corresponds to highest priority queue and 1 corresponds to lowest priority queue
Repeat until all tasks are allocated or all servers are fully loaded.

FIG 2. PROPOSED WORKING MODEL.

VI. CONCLUSION AND FUTURE WORK

This paper is proposing an enhanced priority based HTV load balancing algorithm to perform the effective and reliable resource allocation. In this, the proposed algorithm is calculating the load and performance factor of each virtual machine and then allocating the incoming tasks to various virtual machines according to their priorities. According to the proposed algorithm we conclude that by considering the priorities of the tasks, our algorithm will increase the throughput and performance.

Future work includes the implementation of the proposed work in the existing Cloud computing architecture for effective utilization of resources and better performance.

REFERENCES

1. Jitendra Bhatia , Tirth Patel , Harshal Trivedi , Vishrut Majmudar , “HTV Dynamic Load Balancing Algorithm for Virtual Machine Instances in Cloud”,978-0-7695-4931-6/12,2012 IEEE.
2. He-Sheng WU , Chong-Jun WANG and Jun-Yuan XIE , “ TeraScaler ELB-an Algorithm of prediction-based elastic Load Balancing Resource Management in Cloud Computing”, 978-0-7695-4953-1/13,2013 IEEE.
3. Dr. Hemant S.Mahalle ,Prof. Parag R. Kaveri , Dr. Vinay Chavan, “Load Balancing On Cloud Data Centres”, 2013 IJARCSSE.
4. Steffen Heinzl , Christoph Metz, “Towards a Cloud – ready Dynamic Load Balancer based on the Apache Web Server”,978-0-7695-5002-2/13 , 2013 IEEE.
5. Neeraj , MS.Alankrita Aggarwal, “Load Balance Based on Scheduling and Virtual Machine”, 2013,IJARCSSE.
6. Amandeep Kaur Sidhu , Supriya Kinger, “Analysis of Load Balancing Techniques in Cloud Computing”,vol 4 , March-April,2013 , IJCT.
7. Sewook Wee, Huan Liu , “Client – side Load Balancer using Cloud”,978-1-60558-638-0/10/03, March 2010 ACM.
8. Shu-Ching Wang , Kuo-Qin Yan , Wen-Pin Liao and Shun-Sheng Wang, “Towards a Load Balancing in a Three-level Cloud Computing Network”,978-1-4244-5540-9/10, 2010 IEEE.