# Cloud Resource Management using Load Balancing Algorithm in Cloud Computing Environment

**Bhammar Bhikhu[1] Prof. M.B. Chaudhari[2] Ramachandran[2]**
[1]P.G. Student [2]Associate Professor
[1,2]Department of Computer Engineering
[1,2]Govt. Engineering College, Gandhinagar, Gujarat, India

*Abstract*— Recently, cloud computing is becoming increasingly popular in terms of internet technology .and provide client with wide range of Scalable, distributed, virtualized hardware and software resource. Based on client request that request execution there are challenges of load balancing in cloud computing. Nowadays applications ranging from simple web site to large scale e-commerce applications are being deployed to cloud infrastructure. As applications, especially data-intensive applications, often need to communicate with related data frequently, the overall application performance would be affected by network input and output performance significantly. So proper scheduling of virtualized resource required for providing efficient service. That can be done by efficient load balancing algorithm. In this work we present some modification of algorithm that provides batter solution in terms of response time.

***Key words:*** Load Balancing, Cloud Computing, Virtualization

## I. Introduction

Cloud computing service is a it based service. provide infrastructure, software application on over internet. Cloud computing provide virtualized resource sharing over internet based on pay as you go manner. For incoming request in cloud computing proper allocation of VM is a task of load balancing algorithm. So load balancing is important to allocation of VM and for response time.

In this work we use live migration and load balancing. If request come at datacenter then it will check hash table here we maintain two separate hash table for AVAILABLE and BUSY if VM available then it will directly allocate VM to that request. But if VM not available then we use concept of live migration. In both cases after executing request it make it AVAILABLE in hash table for another request. The reminder of this paper is organized as follows Section II Related work, Section III Proposed Methodology Details, Section IV Result analysis, Finally Conclusion and Future work are outlined in this section.

## II. Related Work

Load balancing is a bigger problem in datacenter. In this section throws light on exiting work in load balancing in cloud computing.

In [2] dynamic nature of load balancing algorithm required. For a maximum utilization of resource for a particular request gives a better response time. Here author suggest adaptive load balancing algorithm (ALBA). Algorithm use intelligent agent to keep track of record of load on virtual machine and also keep information of status of every virtual machine, resource available on VM and time and response time and queue lenth of VM. Based on this properties' request analyzer (RA) can check request for a particular resource.

In [1] use ”LinearPredicting Method” (LPM) and ”Flat Period Reservation-Reduced Method” (FPRRM) to get some useful information from the utilization log, it useful for predicting of resource efficiently for a particular request. Provide better response time during rapid growing period.

In [4] executions analysis of algorithms shows that change of millions instructions per second (MIPS) will affect the performance of the algorithm. Increase in MIPS decrease the response time.

In [5] use migration policy for load balancing. Algorithm check CPU utilization of virtual machine is equal to or greater than 80% then that value prevents machine being overloaded. If virtual machine CPU utilization is greater than 80% than that machine are added to overloaded matrix. Then algorithm check bias matrix which is used to maintain list of virtual machines which are reserved for a particular task. If virtual machine available in bias matrix than it will allow virtual machine migration for load balancing.

In [7] better utilization of resource. In this paper algorithm detect underutilized and over utilized host then place new vm for better load balancing. So it is give better response time and better utilization of resource.

## III. Proposed Methodology Detail

The proposed algorithm is using the concept of both Active Monitoring Load Balancing Algorithm and Throttled Load Balancing Algorithm. In this algorithm we create two lists (which is basically hash table) in which we are storing the key of the each virtual machine and associated status (i.e. either as “AVAILABLE” or “BUSY”) for each virtual machine (VM). When a request arrives from Userbase then it checks the available list of VM‟s, if any VM is found available in this list, it allocates the request to that VM and set status of that machine as “BUSY”, otherwise, if no VM is free, request waits in queue till any process completes its execution and VM changes its status as “AVAILABLE”.

If all VM are BUSSY then overloaded condition occur then here we detect overloaded and under loaded condition work well in underloded condition but in overloaded condition we have to place new virtual machine to host for better utilization of resourse. After executing request make AVAILABLE to that VM for further execution.

Prop_Algorithm_Load_balancing
{
Initialize all VM allocatin status to AVAILABLE in the VM state list
Current allocation count=Syncronized All alocated VM
While(New requests are received by the datacenter controler)

```
Do
{
Data centercontroler queues the requests.
If(current allocation.count.size()<VMstateslist.size())
{
Data center controler remove request from beginning og queue
VMid=key
VMstatelist.put(VMid.VirtualMachine.Busy)
Allocate VM Process the request
VMstatelist.put(VMid.VirtualMachine.Available)
}
}
}
}
```

For overloaded condition

In cloud computing environment data could be distributed in several blocks. These blocks could store in logical or geographical distance.

$$D_{i,j} = \begin{pmatrix} d_{1,1} & d_{1,2} & \cdots & d_{1,j} \\ d_{2,1} & d_{2,2} & \cdots & d_{2,j} \\ \cdots & \cdots & \cdots & \cdots \\ d_{i,1} & d_{i,2} & \cdots & d_{i,j} \end{pmatrix}$$

Here, Row number *i* represents the total number of data storage node and Column number *j* represents the total number of data which will be accessed by the Application Now, size of the application data could be determined by the following equation:

$$\text{Size } j = \sum_{c=1}^{i} d\ c,j \qquad (1)$$

Following formula define the network speed between physical machine which host the VM and the data storage node:

$$S = \text{Data transfer rate}(s, \Delta t) \qquad (2)$$

Here, *S* represents the size of the package and, *Δt* represents the package transfer time slot The matrix *S n,m* can be obtained as follows:

$$S_{n,m} = \begin{pmatrix} s_{1,1} & s_{1,2} & \cdots & s_{1,m} \\ s_{2,1} & s_{2,2} & \cdots & s_{2,m} \\ \cdots & \cdots & \cdots & \cdots \\ s_{n,1} & s_{n,2} & \cdots & s_{n,m} \end{pmatrix}$$

Inverse of the speed between the physical machine and data storage node. Other matrix is Requirement matrix which represents the resource requirement of the VMs to be placed. Resource requirement of the VMs are given along CPU, Memory and Network bandwidth dimension. The Requirement matrix R n,d is defined as follows:

$$R_{n,d} = \begin{pmatrix} r_{1,1} & r_{1,2} & \cdots & r_{1,d} \\ r_{2,1} & r_{2,2} & \cdots & r_{2,d} \\ \cdots & \cdots & \cdots & \cdots \\ r_{n,1} & r_{n,2} & \cdots & r_{n,d} \end{pmatrix}$$

Where each element *r i , j* indicates the requirement of VMi along the dimension j,

The data access time matrix T n,i can be calculated using the following formula:

$$T\ n,i = D\ i,j \times S\ n,m \times R\ n,d \qquad (3)$$

Now, the total resource capacity of the physical machine is defined as set *C*:

$$C = \{P, M\} \qquad (4)$$

Where, *P* indicates the processor capacity of the physical machine and *M* indicates the memory capacity of the physical machine. The set

$$\text{Coccupied} = \{\text{PVMi}, \text{M VMi}\} \ (5)$$

Indicates the resource requirement of *VMi.* Therefore, the available computing resources set can be represented as:

$$\text{Cavailable} = \{\text{P available}, \text{M available}\}$$
$$= \{P - \sum_{i=1}^{n} \text{Pvmi}, M - \sum_{i=1}^{n} \text{Mvmi}\} \qquad (6)$$

Where, n indicates the number of running VMs. Arriving Computation resource requirement of the VM defined as follows:

$$\text{CVMarriving} = \{\text{PVMarriving}, \text{M VMarriving}\} \quad (7)$$

VM would be placed on the PM if the available resource on the physical machine satisfies the following constraints:

$$P - \sum_{i=1}^{n} \text{Pvmi} > \text{P vmi} \ (8)$$

And

$$M - \sum_{i=1}^{n} \text{Mvmi} > \text{M vmi} \ (9)$$

The new available computing resource capacity after the arriving VM is placed can be obtained by the following formula:

C available = {P available − PVMi, M available − MVMi }

## A. *Input:*

Data storage matrix D i, j, Network status matrix S n, m, Resource requirement matrix Rn, d, A coming application requested dataset.

## B. *Output:*

Allocation of VMs

1) Calculating the data transfer time matrix T n,i
2) Traverse all the column in the matrix T n,i to find Minimum $\sum_{y=1}^{n} Tx, y$, && C available > CVMarriving Return x;
3) Allocating the arriving VM on the host h x

If (VM allocation is succeed)
C available= { P available – PVMi, M available − MVMi }
End if

# IV. Result Analysis

## A. *Average Response Time*

In a below graph we compare two exiting algorithm and proposed algorithm, from graph we can say that proposed algorithm provide better response time

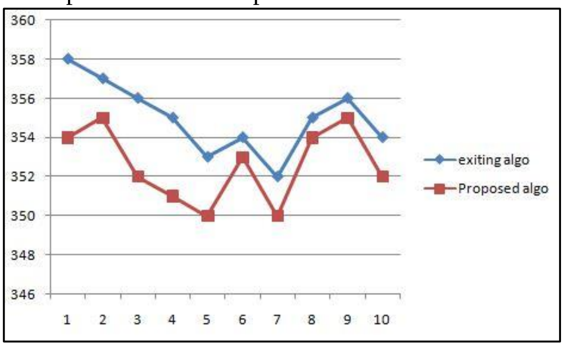


Fig. 1: Graph of Average response time

## B. *Data Center Processing Time*

In a below graph we compare two exiting algorithm and proposed algorithm , from graph we can say that proposed algorithm provide better data center processing time.

 

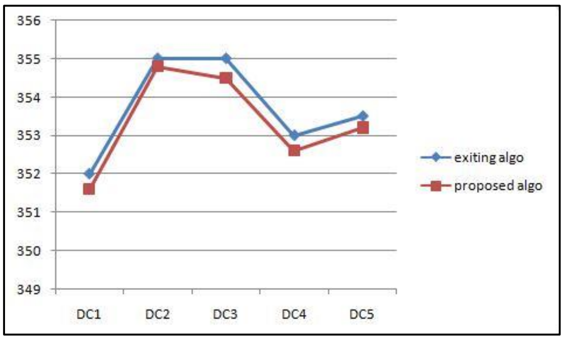

Fig. 2: Graph of Data Center processing Time

## V. CONCLUSION AND FUTURE WORK

In this work proposed system provide batter response time and data center processing time. Because when request comes at data center then it assign VM to that request so it provide better utilization of resource. In future work we can reduce energy consumptin in data center.

## REFERENCES

[1] Yuxiang Shi, Xioahong Jiang and Kejiang Ye, “An Energy Efficient Scheme For Cloud Resource Provisioning Based On CloudSim”, 2011 IEEE International Conference On Cluster Computing.
[2] Manisha Malhotra and Aarti Singh, “Adaptive Framework For Load Balancing to Improve the Performance of Cloud Environment” 2015 IEEE International Conference on Computational Inteligence and Communication Technology.
[3] Mikhail Grasimenko, Dmitri Moltchanov, and Raman Florea , “Cooperative Radio Resource Manazement in Hetrogenous Cloud Radio Access Networks”, IEEE Communication Magazine, February 2015.
[4] Soumya Ray and Ajanta DE Sarkar, “ Execution Analysis of Load Balancing Algorithms in Cloud Computing Environments”, International Journal of Cloud Computing Services and Architecture, vol.2, No.5, October 2012
[5] N.S Raghva and Deepti Singh, “Comparative Study on Load Balancing in Cloud Data center Using Load Balancing,” Open Journal of Mobile Computing and Cloud Computing Vol. 1, NO 1,August 2014.
[6] Ankiya Sharma and Upidar Pal Singh, “Energy Efficiency in Cloud Data center Using Load Balancing”, International Journals of Computer trends and Technology , Volume 11 , No. 5,August 2014.
[7] Mohammad Sadegh Mozafari and gholamhossein Dasthaibyfard , “Novel resource allocation algorithm to performance and energy efficiency in cloud computing “ , Springer, June 2014

 