International Journal of Advanced Technology in Engineering and Science
Vol. No.3, Issue 11, November 2015
www.ijates.com
ijates
ISSN 2348 - 7550

LOAD BALANCING IN CLOUD COMPUTING

Neethu M.S

[1]*PG Student , Dept. of Computer Science and Engineering, LBSITW (India)*

ABSTRACT

Cloud computing is emerging as a new paradigm for manipulating, configuring, and accessing large scale distributed computing applications over the network. Load balancing is one of the main challenges in cloud computing which is required to distribute the workload evenly across all the nodes. Load is a measure of the amount of work that a computation system performs which can be classified as CPU load, network load, memory capacity and storage capacity. It helps to achieve a high user satisfaction and resource utilization ratio by ensuring an efficient and fair allocation of every computing resource. Proper load balancing aids in implementing fail-over, enabling scalability, over- provisioning, minimizing resource consumption and avoiding bottlenecks etc. Many algorithms and approaches are proposed by various researchers throughout the world, with the aim of balancing the overall workload among given nodes, while attaining the maximum throughput and minimum time. The main aim of this paper is to discuss some of the existing load balancing algorithms in cloud computing environment.

Keywords: Cloud Computing , Load Balancer , Load Balancing , Virtual Machine (VM) , Cloudlet.

I. INTRODUCTION

Cloud Computing is a new trend emerging in IT environment with huge requirements of infrastructure and resources. Cloud provides stretchy software as service (SaaS) , Platform as Service (PaaS) , Infrastructure as Service (IaaS). The cloud computing is an internet based enlargement in which dynamically accessible and frequently virtualized assets are provided as a service over the internet has become a substantial dispute. Private and Public are the two types of the cloud. There are supplementary service that are offered from cloud apart from the basic cloud service.

The Cloud delivers several resources that are flexible, scalable, secure, and available while saving money and time. Cloud solutions are simple and they don’t require long term contracts and are easier to scale up and down as per the demand. Prefect planning and migration services are needed to ensure a successful implementation. Both Public and Private Clouds can be deployed together to leverage the best of both.

Load balancing is a computer network method for distributing workloads across multiple computing resources, for example computers, a computer cluster, network links, central processing units or disk drives. Load balancing plans to optimize resource use, maximize throughput, minimize response time, and evade overload of any one of the resources. By the use of multiple components with load balancing instead of a single component may increase reliability through redundancy.

Load balancing[1] is one of the central issues in cloud computing. 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 attain a high customer satisfaction and resource utilization ratio , consequently improving the overall performance and resource utility of the system.

The primary purpose of the cloud system is that its client can utilize the resources to have economic benefits. A resource allocation management process is required to avoid under utilization or over utilization of the resources which may affect the services of the cloud. Some of the jobs may be rejected due to the overcrowding for the Virtual Machines(VMs) by the current jobs in the cloud system. Resource allocation and efficient scheduling is an important characteristic of cloud computing based on which the performance of the system is evaluated. Load balancers, on the other hand, make use of an efficient load balancing algorithm in order to send the request to the most suitable virtual machine. Hence various load-balancing algorithms have been proposed in which live migration of load is done in virtual machines to avoid the under utilization and hence improving the and data transfer cost.

A load balancing algorithm which is dynamic in nature does not consider the previous state or behavior of the system, that is, it depends on the current behavior of the system. There were various goals that are related to the load balancing such as to improve the performance substantially , to maintain the system stability etc. Depending on the current state of the system, load balancing algorithms can be categorized into two types they are static and dynamic algorithms**.**

The geographical distribution of the nodes matters a lot in the overall performance of any real time cloud computing systems, especially in case of the large scaled applications like Facebook, Twitter, etc. The complexity of any load balancing algorithm affects the overall performance of the system. Sometimes the algorithm is complex, but is better in terms of throughput and resource utilization. For any load balancing algorithm, it is very important to analyze the traffic flow in real-time scenarios over different geographic regions, and then balance the overall work load accordingly.

II. EXISTING LOAD BALANCING TECHNIQUES

2.1. Fuzzy Based Firefly Algorithm

This algorithm was proposed with a goal to improve performance through partitioning the cloud environment to balance the loads across the partitions. Fuzzy logic is applied to observe the time characteristics during assignment of load across the Virtual Machines(VMs).The architecture of the proposed system [2] is as shown in Fig.1

The primary method involves partitioning cloud based on frequent node allocation to VMs. The nodes are to be classified into various groups like lightly, normal and heavily loaded. Then the set of tasks are entered as input to the load balancer.

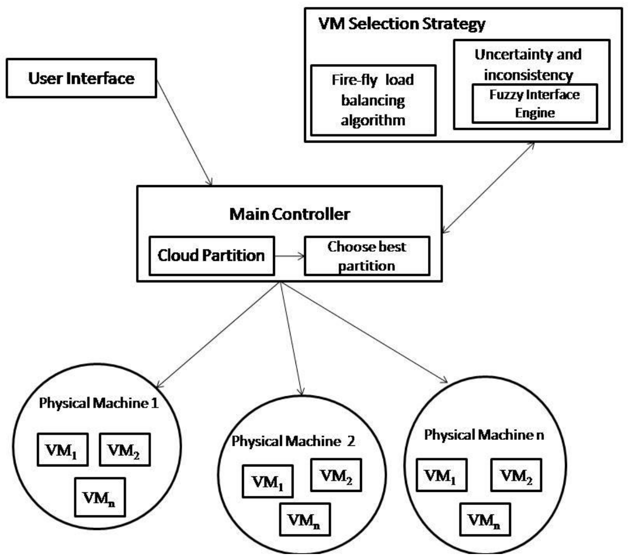

Figure 1. Proposed System Architecture

The characteristic of a firefly is to "attract itself to light" which is being used as the role of a VM for the implementation of the algorithm. A Balance Factor is formulated which is given by :

$$BF=(\alpha*\beta)/(\delta*\varepsilon*\rho)$$

1

where, α –Length of input file

β- Size of file in kilobytes

δ- Processing capacity of the VM supplied with the load

ε- Processing element number

ρ- Speed in number of CPU cycles

The value of balance factor always lies between 0 and 1. As a preliminary fuzzification process the algorithm assigns predefined values like high , medium , low to the tasks that arrives. The fuzzy engine defines the output using a defined rule and on defuzzification process using Smallest of Maximum (SOM) method , the output is obtained. A membership function is calculated :

$$Z\_SOM=MIN(\square Z \text{ € } Z1,Z2)$$

2

where,

Z-Output variable which is the minimum amongst the output variables Z_1,Z_2 etc.

The proposed algorithm can be summarized as follows. $P=\{P_1,P_2,P_3.....P_i\}$ be the number of physical machines being allocated and $M=\{M_1,M_2.....M_j\}$ denotes the allocation of VMs per physical machine. Let the N number of tasks be defined using the set $N=\{N_1,N_2....N_p\}$.Initially the VMs are allocated an ID from M_1 to M_n . Cloudlets are created and each cloudlet is assigned with an ID from N_1 to N_p .Based on an available scheduling scheme the tasks are scheduled. Usually Round-robin scheme is used.

Now firefly algorithm is to be implemented, an M*N matrix calculates the minimum execution time for each task on each VM. Then the load is calculated and a best suited partition or VM has to be selected .The load is then assigned to it. When the threshold exceeds, the control is transferred to the fuzzy logic engine Then the tasks which was already been assigned will be left undisturbed and the remaining will have to undergo the fuzzy logic to generate an output which is balanced.

After the experimental analysis it can be inferred that the execution time required by this algorithm is less, migration and computational cost is less, it has the capacity to handle heavy loads and can balance the amount of loads arrived.

2.2. Honey Bee Behavior Inspired Load Balancing (HBB-LB)

The algorithm is proposed to achieve load balancing by improving through put and minimizing the waiting time of the tasks. By reducing the makespan and response time the reduction in waiting time can be minimized which can result in improving the responsiveness of the VMs. The tasks removed from VM were treated as honeybees.

The mathematical model expresses the set of virtual machines as $VM=\{VM_{1,}VM_{2,}VM_{3}\ldots\ldots VM_{m}\}$ which has to process n tasks given by the set $T=\{T_{1,}T_{2,\ldots\ldots,}T_{n}\}$.Non-preemptive tasks are assigned to the VMs denoted by npmtn. Also the finishing time of the task T_i be given by CT_i. The model is $R|npmtn|CT_{max}$.

Processing time of task T_i on virtual machine VM_j can be denoted as P_{ij}. HBB-LB is a dynamic load balancing technique .Capacity of a virtual machine is given by

$$C_j=pe_{numj} \quad x \quad pe_{mipsj} \quad x \quad vm_{bwj} \qquad 3$$

where , pe_{numj} – Number of processors in VM_j

pe_{mipsj} – Million instructions per second of all processors in VM_j

vm_{bwj} - Communication bandwidth ability of VM_j

Thus the capacity of all virtual machines is give by $C=\sum_{i=1}^{m} C_i$.

The behavioral control structure for HBB-LB [3] can be represented as in the Fig.2 below. The results after implementation shows that the algorithm does not increase any additional overheads. It works well in heterogeneous and non-preemptive environment. The algorithm increases overall throughput and reduces the processing time.

Figure 2. Flow diagram of behavioral control structure for Honey Bee Behavior inspired Load balancing of Tasks.

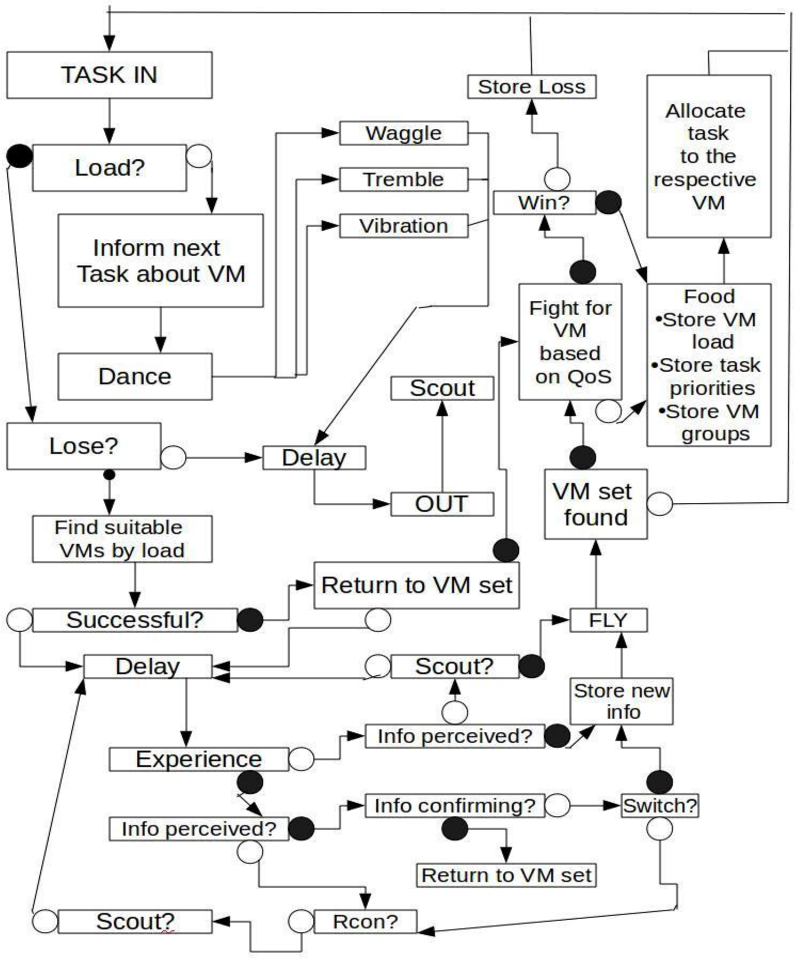

Figure 2. Flow Diagram of Behavioral Control Structure for Honey Bee Behavior Inspired Load Balancing of Tasks.

2.3. Ant Colony Optimization Based Load Balancing Algorithm

Aim of this algorithm is to search for an optimal path between the source of food and colony of the ant on the basis of their behavior. Ants keep record of each and every node that they visits and record that data for future decision making. As a result they deposit pheromones during their movement for other ants to select the next node. Fig.3 shows the possible paths of ant to the source of food. The same concept is used here where the initial pheromone level is a positive integer value τ_o=0.

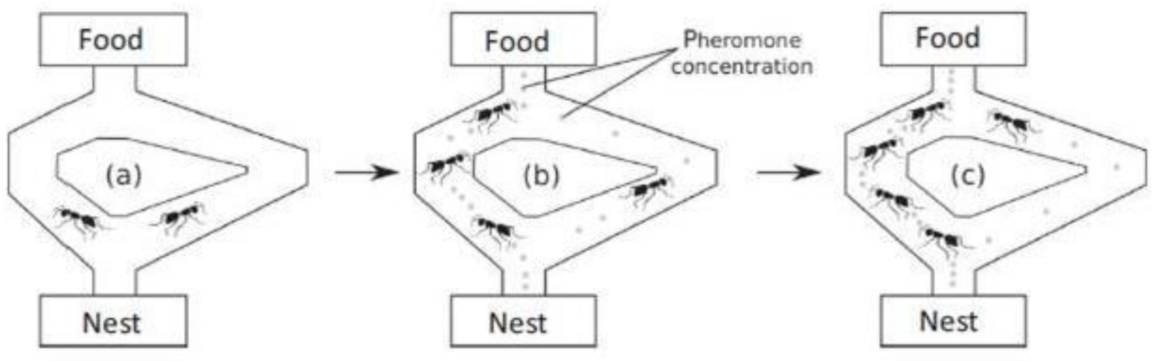

Figure 3. Behavior of ants [4]

A set N=1,2,…n represents the independent jobs and they are distributed and executed on v VMs. On allocation of VMs , the concept is that an each ant works independently and represents a VM "looking" for a host to get allocated. A master table is created which has the details of the loads of each host.

At first a list of all the hosts are created which can be allocated with the VMs. Then a host is selected on the basis of the only if it has processing power, memory and bandwidth greater or equal to that of unallocated VM that is required. On each iteration the information about the loads on a host is entered into the table.

When an ant cannot be complete a given work , which means that it is not allocated to a host of a VM then an exponential back-off strategy is initiated. The load information table acts a pheromone trail. Whenever a host is to be associated to an ant the lightest loaded host is selected from the load information table.

Each time a job is assigned to a VM , a round-robin scheduling was used. As greater the number of serviced users, the throughput was also increased resulting in high parallelism. The algorithm uses indirect communication to exchange information.

2.4 Stochastic-Hill Climbing Algorithm

This algorithm helps in allocating jobs to servers or VMs. Two procedures namely Complete and Incomplete methods are defined to solve the problem of load balancing in an optimized way.

Stochastic-Hill Climbing algorithm[5] is a variant of Incomplete method. When the loop works continuously and the value is increasing it is called "uphill" and stops when it reaches "peak" value. There are main two concepts , a candidate generator and an evaluation criteria. Candidate generator maps one solution candidate to a set of possible successors. Evaluation criteria ranks each valid solution.

The algorithm can be summarized as follows:

Step1:Maintain an index table of Virtual Machine servers(VMs) and the state of the VMBUSY/AVAILABLE .At

the start all VMs are available.

Step 2: A new job arrives in the cloud.

Step 3: Generate query for the next allocation.

Step 4: Generate a VM id randomly.

Step 5: Parse the allocation table from to get the status of the particular VM.

If the VM is found unallocated:

Step 5a: Return the VM id.

```
        Step 5b: Send the request to the VM identified by that id.
        Step 5c: Update the allocation table accordingly.
        If the VM is found to be allocated:
        Step 5d: Use a random function to generated a random VM.
        Step 5e: Select the VM for allocation to the job with a probability such  that this VM
                 will be able to handle the job efficiently.
        Step 5f: Keep account of performance of the VM if it does not perform according to
                 expectation (cost value) decrease its probability for assignment in next iteration.
        Step 5g: Update the allocation table accordingly.
Step 6: When the VM finishes processing the request, and the response cloudlet is received.
        Generate a notification of VM de-allocation.
Step 7: Continue from Step 2 for next allocation.
```

The overall average response time for the algorithm was experimentally found to be less.

2.5 Genetic Algorithm (GA) based Load balancing

The Genetic algorithm based load balancing is performed by minimizing the makespan of tasks. A basic GA is composed with operations selection, genetic operation and replacement. The principle of working of the proposed algorithm can be described using Initial generation ,Crossover and Mutation.

Initial generation is the process of encryption of the possible solutions into binary strings of fixed length.From this an initial population is selected called chromosomes.Crossover is the process of selecting the best pair individual. The available pool of individual chromosomes undergoes a random single point crossover and thus generates a new pair of individual. Mutation is performed by selecting a probability of 0.05.The bits are toggled form 0 to 1 and 1 to 0 and forms a new output for crossover to be performed.

The GA based load balancing algorithm[6] can be summarized as follows:

```
Step1: Randomly initialize a population of processing unit after encoding them into binary strings [Start].
Step2: Evaluate the fitness value of each population using equation for cost function [Fitness].
Step3: While either maximum number of iteration are exceeded or optimum solution is found
       Do:
       Step3(a): Consider chromosome with lowest fitness twice and eliminate the chromosome
                 with highest fitness value to construct the mating pool [Selection].
       Step3(b): Perform single point crossover by randomly selecting the crossover point to
                 form new offspring [Crossover].
       Step3(c): Mutate new offspring with a mutation probability of (0.05) [Mutation].
       Step3(d): Place new offspring as new population and use this population for next round of
                 iteration [Accepting].
       Step3(e): Test for the end condition [Test].
Step4: End.
```

Summarized Review of Algorithms

Algorithms	Static Environment	Dynamic Environment	Centralized Balancing	Distributed Balancing	Hierarchial Balancing
Fuzzy based Fire-Fly	No	Yes	No	Yes	No
Honey Bee Inspired	No	Yes	No	Yes	No
Ant Colony Optimization	No	Yes	No	Yes	No
Stochastic Hill Climbing	No	Yes	No	Yes	No
Genetic Algorithm based	No	Yes	Yes	No	No

III. CONCLUSION

Cloud computing provides everything to the user as a service over network. . In the cloud storage, load balancing is a key issue. It helps in proper utilization of resources and hence in enhancing the performance of the system. This paper gives the summarized review on existing scheduling algorithm as in Table on the basis of the environment for which it was developed. Most of the proposed algorithms were implemented in Java language through CloudSim toolkit. There are many above mentioned algorithms in cloud computing which consist many factors like scalability, better resource utilization, high performance, better response time. The various load balancing algorithms are also being compared here on the basis of different types of parameters.

REFERENCES

[1] Suriya Begum, Dr. Prashanth C.S.R, Review of load balancing in cloud computing, IJCSI International Journal of Computer Science Issues, Vol. 10, Issue 1, January 2013.

[2] A. Vouk, Cloud computing- issues, research and implementations, Information Technology Interfaces, 2008, 31–40.

[3] N. Susila, S. Chandramathi, Rohit Kishore, A Fuzzy-based Firefly Algorithm for Dynamic Load Balancing in Cloud Computing Environment, Journal of Emerging Technologies in Web Intelligence,6(4), IEEE November 2014 , 435-440.

[4] Dinesh Babu .L.D, P.Venkata Krishna, Honey Bee inspired load balancing of tasks in cloud computing environment ,Applied Soft Computing, 13, ,Elsevier 2013, 2292-2303.

[5] Elina Pacini,Cristian Mateos,Carlos Garcia Garino, Balancing throughput and response time in online scientific clouds via Ant Colony Optimization , Advances in Engineering Software ,8,Elsevier 2015,pp.31-47.

[6] Brototi Mondala, Kousik Dasgupta, Paramartha Dutta, Load Balancing in Cloud Computing using Stochastic Hill Climbing-A Soft Computing Approach, Procedia Technology ,4,Elsevier 2012,pp.783-789.

[7] Kousik Dasgupta, Brototi Mandal, Paramartha Dutta, Jyotsna Kumar Mondal, Santanu Dam,A Genetic Algorithm (GA) based Load Balancing Strategy for Cloud Computing, International Conference on Computational Intelligence: Modeling Techniques and Applications (CIMTA) ,Elsevier,2013.