ISSN: 2321-7782 (Online)
Volume 3, Issue 3, March 2015
International Journal of Advance Research in Computer Science and Management Studies
Research Article / Survey Paper / Case Study
Available online at: www.ijarcsms.com

Optimum Load Balancing of Cloudlets Using Honey Bee Behavior Load Balancing Algorithm

Obaid Bin Hassan[1]
Department of Computer Science and Engineering
B.S.Abdur Rahman University
Vandular Chennai-48-India

A Sarfaraz Ahmad[2]
Department of Computer Science and Engineering
B.S.Abdur Rahman University
Vandular Chennai-48-India

Abstract: ***Cloud Computing enables to compute resources as a utility and consumption of computing resources. In Cloud Computing we deploy groups of remote servers and software networks that allow centralized data storage and online access to computer services or resources. Load balancing has become one of the key issues with the rapid growth of web traffic and the services available under cloud environments. It is the job of Load Balancer to distribute the load among different virtual machines so that all the nodes get equally loaded. To attain the maximum throughput and minimum time various researchers throughout the world proposed many algorithms and approaches. In this paper we propose a solution for the existing load balancing algorithm "Honey Bee Behavior Inspired Load Balancing Algorithm" to provide optimum balancing of tasks in cloud environments.***

Keywords: Cloud computing, load balancer, foraging behavior, remote server, scheduler.

I. INTRODUCTION

In cloud computing technology the data and applications are maintained using the internet and central remote servers. Over the last few years Cloud Computing has been gaining immense popularity where user can pay (as you use) for software, hardware,

Infrastructure and computational resources as per user basis. Cloud computing provides a flexible and easy way to store and retrieve huge data without worrying about the hardware needed. The cloud users have been growing exponentially. Therefore, Scheduling of virtual machines in the cloud becomes an important issue to analyze. In cloud, users can either submit their jobs for computational processing in order to complete the task or leave their data in cloud for storage. It is the job of Cloud Scheduler to schedule the tasks in a cloud environment in such a way that the cloud provider can gain maximum benefit for his service.

There are usually two types of scheduling: Preemptive and non-preemptive scheduling. Tasks are usually assigned with priorities. Sometimes it is necessary to run a certain task that has higher priority before another task that is already in running state. Therefore, the running task is interrupted for some time and resumed later when the priority task has finished its execution. This is called Preemptive Scheduling. In non-preemptive scheduling, a running task is executed till its completion. It cannot be interrupted.

In order to utilize the available resources fully, the scheduler should do the scheduling process efficiently. A virtual machine can be assigned with more than one task that runs the tasks simultaneously. For such kind of environments the load should be properly balanced in all virtual machines i.e., one virtual machine should not be heavily loaded as the other virtual machine will remain idle and/or under loaded. In this case, it is the scheduler to do the balancing of loads across the different machines.

To improve the response time of user's submitted applications so that there should be maximum utilization of available resources; we make use of load balancing algorithms. Load balancing methods aims to speed up the execution of applications by removing tasks from over loaded VMs and assigning them to under loaded VMs (figure. 1).

Load balancing can affect the overall performance of a system. Load balancing methods can be classified in two different ways: Static Load Balancing Algorithms and Dynamic Load Balancing Algorithms.

Static Load Balancing Algorithms: Static load balancing algorithms distribute the work among processors prior to the execution of the algorithm i.e., the distribution of work load is done at compile time when resource requirements are estimated. Such algorithms are easy to design and implement.

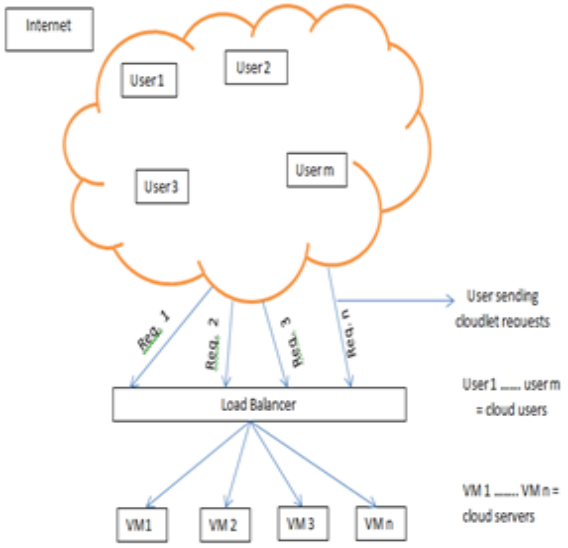

Figure 1 Load Balancing Using Load Balancer

Dynamic Load Balancing Algorithms: Dynamic load balancing algorithms distribute the work among processors during the execution of the algorithm i.e., the distribution of work load among nodes is done at run-time. Algorithms that require dynamic load balancing are somewhat more complicated.

Though load balancing and scheduling seems quite similar but have different meanings. Load balancing is focused on keeping the resources equally busy and to avoid overload of one machine with extra tasks. The advantage is not only in terms of using more computational power but also of spending less time in switching between tasks. Scheduling means deciding the sequence of execution of tasks. Load balancing can be considered as a part of scheduling where time is not considered, with scheduling not considering switching cost but deciding just which task will be executed, and load balancing as optimizing how the selected tasks will be mapped to the resources. In this paper we have considered an existing load balancing algorithm called honey bee behavior inspired load balancing algorithm to overcome its limitations to make the algorithm more efficient.

II. Related Works

Load balancing means to remove tasks from the heavily loaded Virtual Machines and assigning them to the Virtual Machines with low load. Various research papers are there in the field of load balancing to optimize the load across the VMs in cloud environment.

DhineshBabu L.D, et al. [1], considers an algorithm called "Honey Bee Behavior Inspired Load Balancing" which works well by distributing the load equally across virtual machines. This algorithm achieves load balancing so effectively that it reduces the amount of waiting time of tasks in the queue.

Argha Roy, et al. [2], proposes an algorithm in which the fault tolerance in cloud computing using dynamic load balancing technique is taken into consideration. In this method CPU utilization is checked and the load is balanced according to the

utilization of the CPU. With the help of load balancer, the tasks are shifted from the virtual machines with high CPU utilization to the virtual machines with less CPU utilization.

GulshanSoni, et al. [3], proposes a method called "Central Load Balancer" which balances the load in cloud data center. In large scale cloud computing environment this algorithm achieves well load balancing as compared to other load balancing algorithms. In this algorithm every request from user bases arrives at Data Center Controller which are then processed by the Center Load Balancer which also maintains a table that consist of id, states and priority of VMs and finds out the highest priority VMs. The Center Load Balancer then checks the state of the highest priority VM. If its state is available, the id of that VM is returned to the Data Center Controller. If the state is busy, then the next highest priority VM is chosen. After this the Data Center Controller assigns the requests to that VMid that is provided by the Center Load Balancer.

III. Foraging Behavior of Honey Bee

The past few decades have gathered considerable field data on honey bee foraging and searching behavior. In the middle of the 2th century, several studies were performed to analyze how the honey bees communicated exact distances and direction to one another in order to effectively locate the food resources. Karl Von Prisch determined that honey bees perform two distinct dance routines that coincide with two different distance approximations made by the foraging bee. These two dances, the Round dance and the Waggle dance, communicate to the other the approximate distance from the hive to the new resource. The dance that is performed in complete darkness and vertically on a honeycomb is the Round dance. The motion of the Round dance is not straight and is short and direct attracts other foragers, which then learn that the resource is within approximately 50 meters of the hive. According to Von Frisch this dance (Round dance) gives no direction. Because of this, new foragers leave in all directions surrounding the hive in search. This dance is considered favorable because of the short distances travelled. When the resource is at a distance more than 50 meters the Waggle dance is performed. This dance is performed either on a vertical surface or on a horizontal surface by the returning forager.

The bee changes its orientation relative to the sun in order to determine distance and direction. Any deviation from this point gives the new foragers the angle towards which they should pursue. This theory of recruitment involves individual foragers indicating their floral resource in the hive by body movements which can be read by other workers which then go out in the proper direction and the proper distance to locate the same floral path.

According to Johnson and Neigh, Honey Bees which are the social insects, collective decisions are made via feedback based on positive and negative signals, which means that the communication among honey bees through dance (Round dance or Waggle dance) gives an idea to the waiting bees in the nest about a potential food source, its distance from the bee hive, the direction etc. In addition, honey bees also use tremble and vibration dance.

IV. Flow Chart for Load Balancing using Honey Bee Foraging Behavior

The honey bee behavior load balancing algorithm is completely inspired by the foraging behavior of honey bees. When overloaded virtual machine is found, then the task is to be removed to the under loaded virtual machine to balance the load. Here we have two situations i.e., either it finds the VM or it may not find the suitable VM. When it finds a proper VM, a positive signal is generated and when it fails to find suitable VM, a negative signal is generated. Also the task has to find best among the VMs when it finds more than one VM (set of VMs) which is done based on the QoS criteria called task priority. This means that the task finds the VM which has a less number of tasks with same kind of priority. When the task finds the suitable VM, the task is allocated to the respective VM found and the details are updated.

In case the task does not find a suitable VM, it goes to the waiting state where it gets experienced with the information updates by other tasks (figure. 2). The process of finding the proper VM for the waiting task starts again once it confirms the information and will fight with other competing tasks to find its most suitable VM and gets allocated to it. The information is updated after the task is being allocated to the suitable VM. As the new tasks arrive, the process of finding the suitable VM for

each task to balance the load continues until all the tasks are allocated to VMs and the system is properly balanced based on the load as well as priorities.

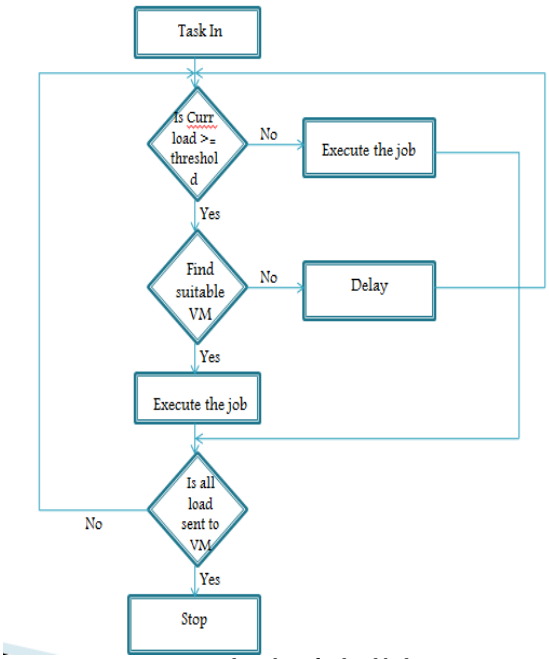

Figure 2 Flowchart for load balancing

V. COMPARISON OF SOME LOAD BALANCING TECHNIQUES

Different load balancing techniques discussed by various researchers has been presented in table I. the comparison is based on some of the performance parameters. The performance parameters on the basis of which the existing load balancing techniques are measured are as:

(i) Throughput: Throughput means the number of successfully executed tasks. For high system performance, high throughput is necessary.

(ii) Overhead: Overhead is the estimation of extra cost involved while making an algorithm to execute efficiently. For an efficient algorithm, there should be very low overhead.

(iii) Fault Tolerance: Fault Tolerance is the capability of an algorithm to perform smooth balancing of load under some failure conditions (e.g., network failure). If any kind of failure occurs, it should not affect the overall system performance. A good load balancing algorithm should be high fault tolerable.

(iv) Migration Time: It is defined as the time required in switching the tasks from one machine to another. For a good load balancing algorithm, migration time should be less.

(v) Response Time: The time between sending a request and receiving its response is called as the response time. As less the response time of an algorithm is, the more efficiently it is going to operate to increase the performance of the system.

(vi) Resource Utilization: This is to ensure that the resources that make up the system are properly utilized in such a way that no tasks should remain idle or keep waiting for the resource to get executed.

(vii) Scalability: Scalability means to increase the number of machines in a system. By increasing the number of machines, the algorithm should perform smoothly as with the lesser number of machines. That is the performance of the algorithm should remain uniform upon increasing or decreasing the number of machines.

Table I: Comparison of Some Existing Load Balancing Techniques.

Algorithm	*Advantages*	*Disadvantages*
Round Robin	High throughput. Less migration time. Optimum resource utilization. Scalable.	More overhead. Less fault tolerant. High response time.
Throttled	Low overhead. Fault tolerant. Optimum resource utilization. Scalable.	Fewer throughputs. High migration time. High response time.
Honey Bee Foraging	Low overhead. Less migration time. Less response time. Optimum resource utilization.	Fewer throughputs. Less fault tolerant. Less scalable.
Max-Min	High throughput. Less migration time. Optimum resource utilization.	More overhead. Less fault tolerant. High response time. Less scalable.
Min-Min	High throughput. Less migration time. Optimum resource utilization.	More overhead. Less fault tolerant. High response time. Less scalable.

VI. PROPOSED WORK

The honey bee behavior load balancing algorithm achieves well balanced load across virtual machines by maximizing the throughput and minimizing the response time. This algorithm balances the priorities of tasks on the machines in such a way that the amount of waiting time of the tasks in the queue is minimal. In spite of this, there are some key issues related to honey bee behavior load balancing algorithm:

- Low priority task stay continuously in the queue.
- Less fault tolerant.
- Less scalable.

Our approach is to overcome one of the issues to make the algorithm more efficient.

The tasks having higher priorities are given the first preference and hence are executed first which results in starvation for the tasks with lower priority as they have to stay back in the waiting queue continuously for their turn when more and more tasks with higher priorities arrive in the queue.

The algorithm is less fault tolerant in the sense that when there is any kind of failure in the system (e.g., network failure), then the algorithm is not able to work efficiently. Under such circumstances, the algorithm may not achieve the proper load balancing or may give the unrealistic output.

The algorithm works well when there is small number of VMs and hence balances the load efficiently. But if the number of VMs is increased, the algorithm starts responding inefficiently and hence may give the improper results.

The above issues can be solved by our proposed approach called the “Root Cause Analysis” approach by which we first analyze why such problems arise and when they arise (i.e., at what point the problem takes place). As the name suggests, Root Cause Analysis approach first tries to find the actual cause or root of the problem which is very essential to find in order to provide specific solution to solve the problem. Root Cause Analysis is the task of identifying the root causes and the corresponding components that they affect. If the root cause of a particular problem can be identified, then the problem can be fixed and the overall risk and threat associated with it will be reduced.

VII. CONCLUSION

In this paper optimum load balancing of cloudlets using Honey Bee Behavior load balancing algorithm has been proposed to provide an efficient utilization of resources in cloud environment. In this algorithm the priorities of tasks is also taken into consideration that have been removed from heavily loaded virtual machines. These removed tasks from the overloaded virtual machines are treated as honey bees which in turn update the information on the respective virtual machines. Also we looked at the key issues of the proposed algorithm which if overcomes makes the algorithm more efficient. We proposed one solution called "Root Cause Analysis" approach which will identify the root causes and the corresponding components that they affect. Once identified, the problem can be fixed then and the overall risk and threat associated with it will be reduced.

References

1. DhineshBabu L.D., P. Venkata Krishna, Honey Bee Behavior Inspired Load Balancing Of Tasks In Cloud Computing Environments, Applied Soft Computing Journal, Vol. 13, 2013.
2. Argha Roy and DiptamDutta, Dynamic Load Balancing: Improve Efficiency In Cloud Computing, International Journal Of Emerging Research In Management &Technology, Vol. 2, April 2013.
3. GulshanSoni and Mala Kalra, A Novel Approach For Load Balancing In Cloud Data Center, International Advance Computing Conference, 2014.
4. B. Yagoubi, Y. Slimani, Task load balancing strategy for grid computing, Journalof Computer Science 3 (3) (2007) 186–194.
5. A. Revar, M. Andhariya, D. Sutariya, M. Bhavsar, Load balancing in grid environment using machine learning-innovative approach, International Journal of Computer Applications 8 (10 (Oct)) (2010) 975–8887.
6. B. Yagoubi, Y. Slimani, Dynamic load balancing strategy for grid computing, transactions on engineering, Computing and Technology 13 (May) (2006) 260–265.
7. B. Yagoubi, M. Medebber, A load balancing model for grid environment, computer and information sciences, 2007. iscis 2007, in: 22nd International Symposium on, 7–9 Nov, 2007, pp. 1–7.
8. Y. Hu, R. Blake, D. Emerson, An optimal migration algorithm for dynamic load balancing, Concurrency: Practice and Experience 10 (1998) 467–483.
9. M. Randles, D. Lamb, A. Taleb-Bendiab, A comparative study into distributed load balancing algorithms for cloud computing, in: Proceedings of 24th IEEE International Conference on Advanced Information Networking and Applications Workshops, Perth, Australia, April, 2010, pp. 551–556.
10. M. Houle, A. Symnovis, D. Wood, Dimension-exchange algorithms for load balancing on trees, in: Proc. of 9th Int. Colloquium on Structural Information and Communication Complexity, Andros, Greece, June, 2002, pp. 181–196.
11. NanditaGoyal, Dynamic Load Balancing Algorithm For Cloud Computing Using Mobile Agents, International Journal Of Advanced Research In Computer Science and Software Engineering, Vol. 3, December 2013.
12. D.L. Eager, E.D. Lazowska, J. Zahorjan, Adaptive load sharing in homogeneous distributed systems, The IEEE Transactions on Software Engineering 12 (5) (1986) 662–675.
13. H.D. Karatza, Job scheduling in heterogeneous distributed systems, Journal ofSystems and Software 56 (1994) 203–212.
14. R.F. de Mello, L.J. Senger, L.T. Yang, A routing load balancing policy for grid computing environments, in: 20th International Conference on, vol. 1, 18–20 April, Advanced Information Networking and Applications, 2006. AINA 2006.(2006)
15. C. Zhao, S. Zhang, Q. Liu, J. Xie, J. Hu, Independent tasks scheduling based on genetic algorithm in cloud computing, wireless communications. Networking and mobile computing, 2009.WiCom '09, in: 5th International Conference on, 24–26 Sept, 2009, pp. 1–4.
16. N. Malarvizhi, V. RhymendUthariaraj, Hierarchical load balancing scheme for computational intensive jobs in Grid computing environment, in: Advanced Computing, 2009. ICAC 2009. First International Conference on, 13–15 Dec, 2009, pp. 97–104.

AUTHOR(S) PROFILE

Obaid Bin Hassan, is doing M.Tech degree in Computer Computers Science and Engineering from BSA University vandular chennai India, and B-Tech from BGSBU Jammu and Kashmir, India in 2012.

