A Technique Based on Ant Colony Optimization for Load Balancing in Cloud Data Center

Ekta Gupta
Department of Information Technology
MIT, Pune
5ektagupta@gmail.com

Vidya Deshpande
Department of Information Technology
MIT, Pune
vidya.deshpande@mitpune.edu.in

***Abstract*----As a large number of requests are submitted to the data center, load balancing is one of the main challenges in Cloud Data Center. Existing load Balancing techniques mainly focus on improving the quality of services, providing the expected output on time etc. Therefore, there is a need to develop load balancing technique that can improve the performance of cloud computing along with optimal resource utilization. The proposed technique of load balancing is based on Ant Colony Optimization which detects overloaded and underloaded servers and thereby performs load balancing operations between identified servers of Data Center. The proposed technique ensures availability, achieves efficient resource utilization, maximizes number of requests handled by cloud and minimizes time required to serve multiple requests. The complexity of proposed algorithm depends on datacenter network architecture.**

Keywords: Ant Colony Optimization (ACO), Cloud computing, Data Center, Load balancing.

I. INTRODUCTION

Cloud computing is relatively new and emerging technology. Cloud computing is not a total new concept; it is originated from the large-scale Grid environment. It is a technology of computing that is extensively used in today's industry as well as society. Applications runs in the cloud, the user can access it anywhere through an Internet-enabled mobile device or a connected computer and available 7x24x365. With a cloud app, you just open a browser, log in, customize the app, and start using it. It has moved computing and data away from desktop to portable PCs into large data centers. As a large number of requests are submitted by users, the cloud Data Center need not only to finish those requests but also to satisfy the user's service demand. How to balance load among nodes of data center efficiently becomes a key problem to be solved in the cloud environment. Nowadays, many companies offering services to the customer based on the concept of "pay as a service", where each customer pays for the services obtained from the provider. The system, which is incurring a cost for the user should function smoothly and should have algorithms that can continue the proper system functioning even at peak usage hours. So, there is a need of proper load balancing technique which helps to achieve high user satisfaction and resource utilization ratio by ensuring an efficient and fair allocation of every computing resource.

II. LITERATURE SURVEY

A. Join–Idle-Queue

Join-Idle Queue is basically used for large-scale systems [1]. It uses distributed dispatchers by first load balancing the idle processors across dispatchers and then assigning jobs to processors to reduce average queue length at each processor. It effectively reduces the system load, incurs no communication overhead at job arrivals and does not increase actual response time.

A. Biased Random Sampling

Martin Randles [2] investigated a distributed load balancing approach. Here, a virtual graph is constructed, with the connectivity of each node (a server is treated as a node) representing the load on the server. Each server is symbolized as a node in the graph, with each indegree directed to the free resources of the server. Whenever a node does or executes a job, it deletes an incoming edge, which indicates reduction in the availability of free resource. After completion of a job, the node creates an incoming edge, which indicates an increase in the availability of free resource. The walk starts at any one node and at every step a neighbor is chosen randomly. Suppose b is the walk length and t is a threshold value of walk length. A node upon receiving a job, will execute it only if its current walk length is equal to or greater than the threshold value. Else, the walk length of the job under consideration is incremented and another neighbor node is selected randomly. When, a job is executed by a node then in the graph, an incoming edge of that node is deleted.

B. Honeybee foraging

This algorithm is derived from the behavior of honey bees that uses the method to find and reap food. In bee hives, there is a class of bees called the scout bees and the another type was forager bees .The scout bee which forage for food sources, when they find the food, they come back to the beehive to advertise this news by using a dance called waggle/tremble/vibration dance. The purpose of this dance,

978-1-4799-8084-0/14 $31.00 © 2014 IEEE
DOI 10.1109/ICIT.2014.54

TABLE I. Comparisons of existing load balancing algorithms for cloud computing

Technique	Merits	Demerits
Join-Idle-Queue [1]	1. Effectively reduces the system load. 2. Incurs no communication overhead at job arrivals.	1. It cannot be used for today's dynamic-content web services due to the scalability and reliability.
Biased Random Sampling [2]	1.Performs better with high and similar population of resources	1.Performance degrades as population diversity increases.
Honeybee Foraging [2]	1. Performance of the system is enhanced with increased system diversity	1. Throughput is not increased with an increase in system size.
Towards a load balancing in Three level cloud computing network [3]	1.Good utilization of resource (not optimized)	1. The task of every level of node will depend on Upper level Node.
Cloud computing initiative using modified ACO framework [4]	1.Better utilization of resources (not optimized)	1. Fault tolerance is not good. 2. Overhead increases.
Dual Direction Downloading Algorithm (DDFTP) [5]	1.Fast and efficient concurrent technique 2.Reliable download of files.	1. Full replication of data files that requires high storage in all nodes
Exponential Smooth Forecast-based on Weighted Lest Connection (ESBWLC) [6]	1.Improves Weighted Least Connection (WLC)	1.Require more processing time than WLC
Active Clustering [7]	1. The performance of an algorithm can be enhanced by making a cluster of nodes.	1. Performance of algorithm is degraded with an increase in system diversity.

gives the idea of the quality and/or quantity of food and also its distance from the beehive. Forager bees then follow the Scout Bees to the location that they found food and then begin to reap it. After that they return to the beehive and do a tremble or vibration dance to other bees in the hive giving an idea of how much food is left. M. Randles [2] investigated a decentralized honeybee-based load balancing technique that is a nature-inspired algorithm for self-organization. It achieves global load balancing through local server actions. Performance of the system is enhanced with increased system diversity but throughput is not increased with an increase in system size.

C. Towards a load balancing in three level cloud computing network

Wang [3] suggested an algorithm called Load Balancing Min-Min (LBMM). In this algorithm nodes are distributed in three level structures where work is distributed among nodes. It combines OLB (Opportunistic Load Balancing) and LBMM (Load Balancing Min-Min) algorithm. This technique uses Opportunistic Load Balancing algorithm which keep each node busy in the cloud without considering execution time of node. Because of this it causes bottle neck in system. This problem is solved by LBMM Three Layer Architecture. In first level LBMM architecture is the request manager which is responsible for receiving the task and assigning it to service manager, when the service manager receives the request; it divides it into subtask and assigns the subtask to a service node based on node availability, remaining memory and the transmission rate which is responsible for execution the task.

D. Cloud computing initiative using modified ACO framework

Cloud Computing Initiative Using Modified ACO Framework [4] algorithm minimizes the make span (throughput of heterogeneous computing system). This algorithm modified the basic ACO. Modification involved the basic pheromone updation formula. It gives better utilization of resource but overhead is high.

E. Dual Direction Downloading Algorithm (DDFTP)

DDFTP [5] is a dual direction downloading algorithm from FTP server. This algorithm can be also implemented for Cloud Computing load balancing. This is a fast and efficient concurrent technique for downloading large files from FTP server in a cloud environment. DDFTP uses the

concept of processing the files for transfer from two different directions. For example, one server will start from block 0 and keeps downloading incrementally while another server start from block m and keeps downloading in a decrement order. When the two servers download two consecutive blocks, the task is considered as finished and other task can be assigned to the server. As a result, both servers will work independently. The algorithm reduces the network communication between the client and nodes and network overhead.

F. Exponential Smooth Forecast-based on Weighted Lest Connection (ESBWLC)

ESBWLC is a dynamic load balancing algorithm [6] for cloud computing. ESBWLC build the conclusion of assigning a certain task to a node after having a number of task assigned to that service node and getting to know the node's CPU power, memory, number of connections and the amount of disk space currently in used, then ESBWLC predicts which node is to be selected based on exponential smoothing.

G. Active Clustering

Active Clustering [7] is a clustering based algorithm which introduces the concept of clustering in cloud computing. In cloud computing there are many load balancing algorithms available. Each algorithm has its own advantages and disadvantages. Depending on the requirement, one of the algorithms is used. The performance of an algorithm can be enhanced by making a cluster of nodes. Each cluster can be assumed as a group. The principle behind active clustering is to group similar nodes together and then work on these groups. Performance of algorithm is degraded with an increase in system diversity.
The process of creating a cluster revolves around the concept of match maker node. In this process, first node selects a neighbor node called the matchmaker node which is of a different type. This matchmaker node makes connection with its neighbor which is of same type as the initial node. Finally the matchmaker node gets detached. This process is followed iteratively.

Table I show the comparative study of different load balancing.

III. PROPOSED LOAD BALANCING TECHNIQUE BASED ON ANT COLONY OPTIMIZATION (ACO)

The proposed technique of load balancing is based on Ant Colony Optimization (ACO) concept. ACO is inspired from the ant colonies that work together in foraging behavior. In fact the real ants have inspired many researchers for their work and the ants approach has been used by many researchers for problem solving in various areas. This approach is called on the name of its inspiration ACO. The ants work together in search of new sources of food and simultaneously use the existing food sources to shift the food back to the nest. The ants leave a pheromone trail upon moving from one node to another. By following the pheromone trails, the ant subsequently came to the food sources. The intensity of the pheromone can vary on various factors like the quality of food sources, distance of the food, etc. The ants use these pheromone trails to select the next node. A Data Center server is known as node in the proposed system.

A. Pheromone Updation

The ant will use two types of pheromone for its movement. The type of pheromone being updated by the ant would signify the type of movements of the ant and would tell about the kind of node the ant is searching for (i.e. overloaded or underloaded node). The two types of pheromones updated by the ants are as follows:

Foraging Pheromone (FP)
While moving from underloaded node to overloaded node, ant will update FP. Equation for updating FP pheromone is

$$FP(t+1) = (1 - \beta_{eva})FP(t) + \Delta FP \quad (1)$$

Where,
β_{eva} = Pheromone evaporation rate
FP = Foraging pheromone of the edge before the move
FP(t+1) = Foraging pheromone of the edge after the move
ΔFP = Change in FP

Trailing Pheromone (TP)
While moving from overloaded node to underloaded node, ant will update TP. Equation for updating TP pheromone is

$$TP(t+1) = (1 - \beta_{eva})TP(t) + \Delta TP \quad (2)$$

Where,
β_{eva} = Pheromone evaporation rate
TP = Trailing pheromone of the edge before the move
TP(t+1) = Trailing pheromone of the edge after the move
ΔTP = Change in TP

Therefore, the ants use these pheromone trails according to the kind of nodes they encounter. The ants after originating from the head node, by default follow the Foraging pheromone (i.e. searching for overloaded node), and in the process, they update the FP trails according to the formula.

Fig. 1 show the flowchart of proposed load balancing algorithm based on ACO for cloud data center.

After originating from head nodes, ant move to node called as nextnode. This nextnode is randomly selected from the neighbor nodes of head node. The encountered nextnode status can either be overloaded or underloaded. Now this nextnode is became currentnode. Suppose, the load on currentnode is greater than threshold i.e. status of currentnode is overloaded. Now, ant will search for

underloaded node among the neighboring nodes of the currentnode. Here, ant can either get all overloaded neighbors or one underloaded node with minimum load. If

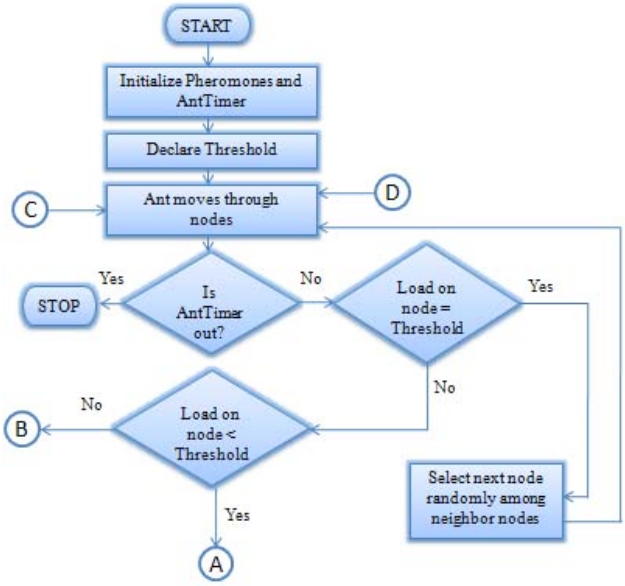

Figure 1. proposed load balancing algorithm based on ACO for cloud

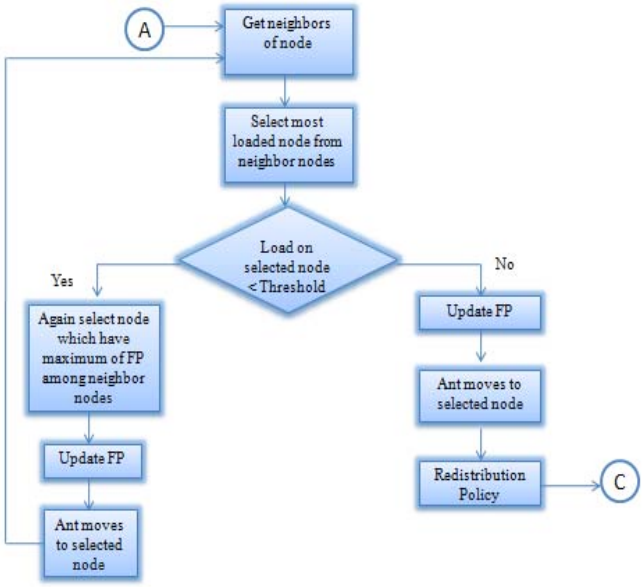

Figure 2. Underloaded → Overloaded

ant get underloaded node then it will move to that node and update TP otherwise, ant will select the node which has minimum TP among neighbor nodes of currentnode and then move to that node and then update TP. Now, the node on which ant is moved, is became a currentnode. Now, redistribution of load is done if and only if currentnode is underloaded. Otherwise, it will again search for the underloaded node among neighbor nodes of currentnode.

Now suppose, the load on currentnode is less than threshold i.e. status of currentnode is underloaded. Now, ant will search for overloaded node among the neighboring nodes of the currentnode. Here, ant can either get all underloaded neighbors or one overloaded node with maximum load.

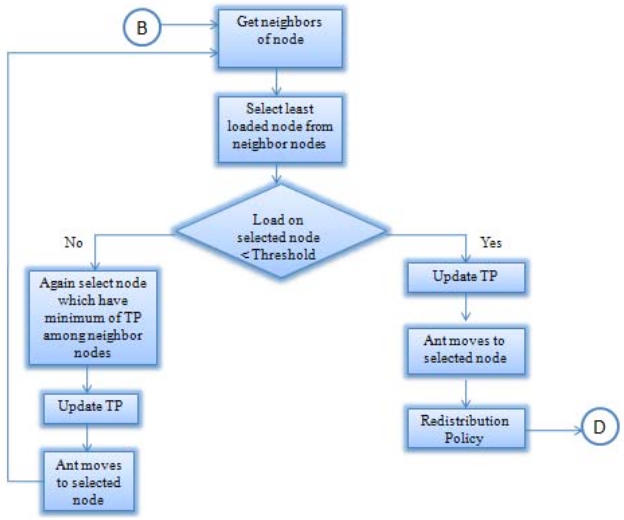

Figure 3. Overloaded → Underloaded

if ant get overloaded node then it will move to that node and update FP otherwise, ant will select the node which has maximum FP among neighbor nodes of currentnode and then move to that node and then update FP. Now, the node on which ant is moved, is became a currentnode. Now, redistribution of load is done if and only if currentnode is overloaded. Otherwise, it will again search for the overloaded node among neighbor nodes of currentnode.

B. The proposed redistribution policy

The main task of ants in the algorithm is to redistribute request among the nodes. The proposed redistribution policy is used to calculate that how many requests should be given to each node which is involved in redistribution. The proposed redistribution policy is explained with the help of following example.

Suppose, there are two nodes N1 and N2 with load of 18 and 27 respectively. Now, add the load of N1 and N2, A = 18 + 27 = 45. Now, divide A by 2, Half = 45 ÷ 2 = 22.5 i.e. 22. Now, subtract Half from A, 45 – 22 = 23. Therefore, N1 will get load of 22 requests and N2 will get load of 23 requests.

IV. SIMULATION AND RESULTS

Simulation is done using NetBeans. There are 36 simulated nodes in Data Center. Head node with ID 15 is fixed because in the simulation of network of servers it has maximum number of neighbors. Suppose, there are 15000 requests are distributed on nodes and calculated threshold is 417. Threshold value depends on number of requests and availability of resource. AntTimer is for 1000ms. Number of ants i.e. threads in system is 8.

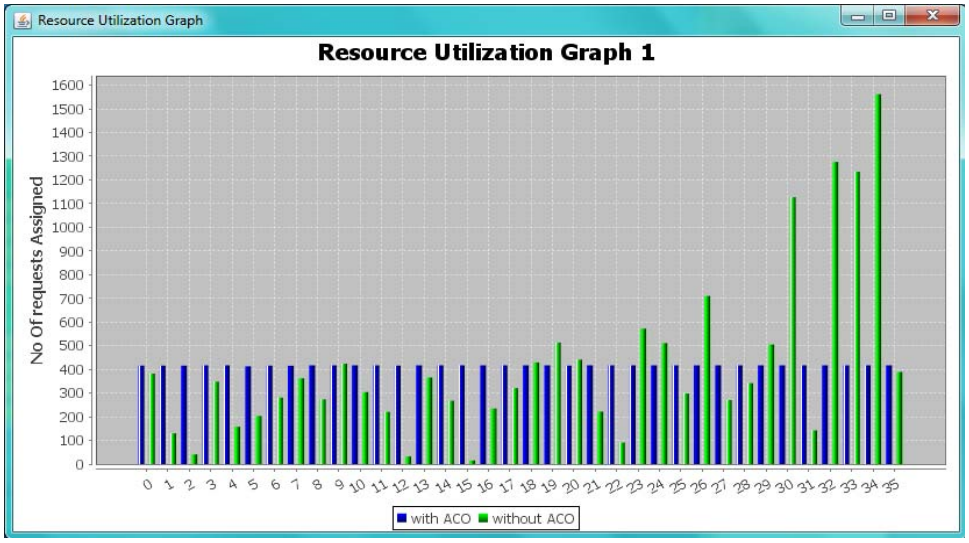

Figure 4. Resource Utilization Graph

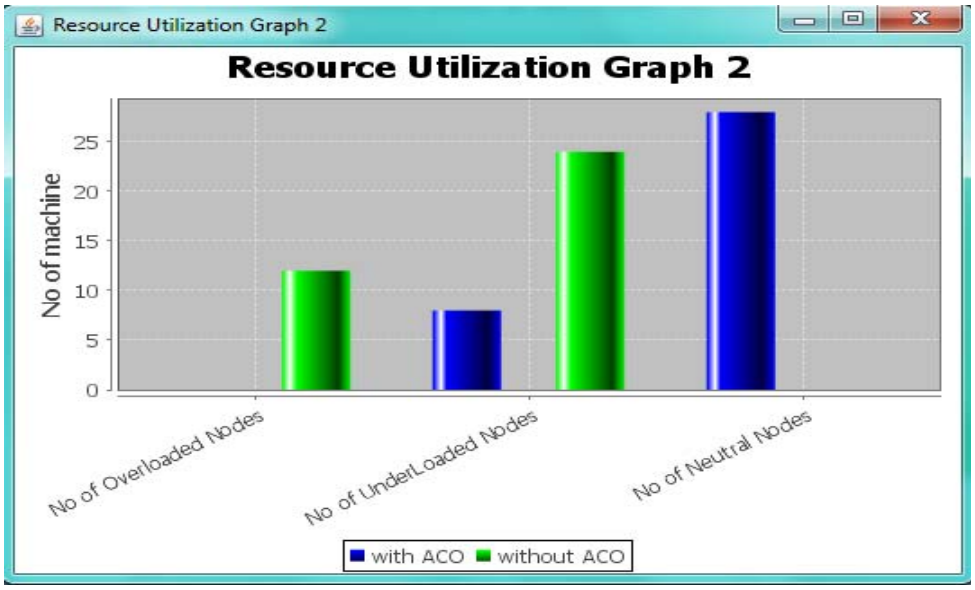

Figure 5. Resource Utilization Graph for Node Type

Fig. 4 shows the bar graph for resource utilization with and without proposed load balancing technique.

Fig. 5 shows the bar graph for resource utilization which shows comparison of number of underloaded, overloaded and neutral nodes in the system with and without proposed load balancing technique.

V. CONCLUSION AND FUTURE SCOPE

As a large number of requests are submitted to the data center, load balancing is one of the main challenges in Cloud Data Center. Existing load Balancing techniques that have been studied mainly focus on improving the quality of services, providing the expected output on time etc. Therefore, there is a need to develop load balancing technique that can improve the performance of cloud computing along with maximum resource utilization. The proposed load balancing technique based on Ant Colony Optimization gives optimal resource utilization. The performance of the system is enhanced with high availability of resources, thereby increasing the throughput. This increase in throughput is due to the optimal utilization of resources. Future work is to implement this technique with the consideration of server's CPU power, memory etc while redistributing load and also to do implementation of algorithm for finding out neighbors of a node with particular Data Center Network Architecture.

REFERENCES

[1] Jayant Adhikari, Sulbha Patil, "Load Balancing the Essential Factor in Cloud Computing", International Journal of Engineering Research & Technology, ISSN: 2278-0181, Vol. 1 Issue 10, December 2012.

[2] Martin Randles, David Lamb, A. Taleb-Bendiab, "A Comparative Study into Distributed Load Balancing Algorithms for Cloud Computing", 2010 IEEE 24th International Conference on Advanced Information Networking and Applications Workshops.

[3]S.C. Wang, K.Q. Yan, W.P. Liao and S.S. Wang, "Towards a Load Balancing in a Three-level Cloud Computing Network", Proceedings of the 3rd IEEE International Conference on Computer Science and Information Technology, pp. 108-113, 2010.

[4] Shagufta khan, Niresh Sharma, "Ant Colony Optimization for Effective Load Balancing In Cloud Computing" International Journal of Emerging Trends & Technology in Computer Science (IJETTCS) ISSN 2278-6856 Volume 2, Issue 6, November – December 2013

[5] Al-Jaroodi, J. and N. Mohamed. "DDFTP: Dual-Direction FTP," 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), IEEE, pp:504-503, May 2011.

[6] Ren, X., R. Lin and H. Zou, "A dynamic load balancing strategy for cloud computing platform based on exponential smoothing forecast".International Conference on. Cloud Computing and Intelligent Systems (CCIS), IEEE, pp: 220-224, September 2011.

[7] Shanti Swaroop Moharana, Rajadeepan D. Ramesh & Digamber Powar," Analysis of Load Balancers In Cloud Computing", International Journal of Computer Science and Engineering (IJCSE) ISSN 2278-9960 Vol. 2, Issue 2, May 2013, 101-108