A Load Balancing Technique for Servers of Datacenter of Cloud using Ant Colony Optimization

[1]Ekta Gupta, [2]Vidya Deshpande

Department of Information Technology
Maharashtra Institute of Technology Pune, India
Email: [1]5ektagupta@gmail.com, [2]vidya.deshpande@mitpune.edu.in

Abstract - In this paper, the technique for load balancing is based on Ant Colony Optimization (ACO). The technique can adapt to the dynamic cloud environment. In this approach, there are two types of pheromones, Foraging Pheromone and Trailing Pheromone. This helps in proper traversing of ant even when the node type (overloaded or underloaded) is changed. Load balancing technique is used to maximize number of requests handled by cloud and to minimize time required to serve those requests. Load balancing technique helps to improve user satisfaction and resource utilization ratio even at peak usage hours.

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

I. INTRODUCTION

Distributed processing, parallel processing and grid computing collectively emerges as cloud computing. Gartner defines cloud computing as "a style of computing where massively scalable IT-enabled capabilities are delivered 'as a service' to external customers using Internet technologies." [7]. There are three keywords in this definition Scalable, Service and Internet. Cloud computing provides services using internet. It provides scalable infrastructure. It helps to adapt changes in demand.

In cloud computing environment, services offered to the customer are based on the concept of "pay as a service", where each customer pays for the services obtained from provider. So, the system which is incurring a cost for the user should function smoothly even at peak usage hours [7].

In the face of a large number of requests which are submitted by users, the cloud data centers 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.

A load balancing technique is needed to avoid the condition where some cloud nodes are heavily loaded while others are idle or doing less work [7]. The proper load balancing technique for cloud improves high user satisfaction. There are number of aims of load balancing technique like maximize number of requests handled by cloud and to minimize time required to serve those requests [7]. This paper proposes an algorithm based on ACO to balance load .The algorithm can adapt to the dynamic cloud environment. The result shows that the proposedalgorithm has better performance and load balancing ability.

II. A LITERATURE SURVEY OF DIFFERENT LOAD BALANCING ALGORITHMS FOR CLOUD

T. Kokilavani et al. [2] has studied the relative performance of Minimum Execution Time (MET). This algorithm assigns job to the resource which has minimum expected execution time without considering the availability of the resource and its current load. He also studied the relative performance of task scheduling algorithm called Min-Min algorithm. The Min-Min algorithm first finds the minimum execution time of all tasks. Then it selects the task with the minimum execution time among all the tasks. Then algorithm assigns the task to the resource that produces the minimum completion time. The same steps are repeated by Min-Min algorithm until all tasks are scheduled. In MAX-MIN algorithm, the machine that has the minimum completion time for all jobs is selected. Then the job with the overall maximum completion time is selected and allocated to that resource. The ready time of the resource is updated. This process of algorithm is repeated until all the unmapped tasks are assigned. The aim of this algorithm is to minimize the waiting time of the larger jobs. Isam Azawi Mohialdeen [3] has studied the performance of the Minimum Completion Time job scheduling algorithm. In this algorithm, selected job will be allocated to the available VM that can offer the minimum completion time taking into account its current load.He also studied the performance of the Opportunistic load balancing algorithm. This algorithm attempts to dispatch the selected job to the available VMs which has the minimum load compared to the other VMs. The idea is to scale the current loads for each VM before sending the job. Then, the VM that has the minimum load is selected to run the job.Jayant

Adhikari et al. [4] have discussed an algorithm called Equally Spread Current Execution Algorithm. The cloud manager estimates the job size and checksfor the availability and capacity of the virtual machine. When the job size and the availableresource size match, the job schedulerimmediately allocates the identified resource to the job inqueue. Manoranjan Dash et al. [5] have discussed an algorithm called Throttled algorithm. Throttled algorithm is totally based on virtual machine. In this client first requesting the load balancer to check the right virtual machine which access that load easily and perform the operations which is given by the client or user. Supriya Kinger et al. [1] have discussed a connection mechanism for Load balancing. Load balancing algorithm can also be based on minimum connection mechanism. When a new connection is dispatched to the node, the number of connection increases. When connection finishes or timeout happens, the number of connection decreases.Martin Randles [6] investigated a distributed and scalable load balancing approach called biased random sampling. This approach uses random sampling of the system domain to achieve self-organization thus balancing the load across all nodes of the system. This technique fails to provide proper resource utilization.He also discussed honeybee foraging algorithm. This algorithm is derived from the natural behavior of honeybees for finding and reaping food.

These 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 algorithm based on ACO gives optimized resource utilization.

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

ACO is a branch of Swarm Intelligence.ACO is inspired from the foraging behavior of ant colonies. In fact the real ants have inspired many researchers for solving the problem in different areas [7]. ACO gives better performance compare to other load balancing algorithms. Inherent parallelism, robustness & scalability along with simplicity of individual agent are the advantages of ant-based systems. While searching for food, the ant deposits a chemical called pheromone trail on the ground. The quantity of pheromone deposited, which may depend on the quantity and quality of the food, will guide other ants to find the food source.

In this proposed algorithm, first a Head node is selected. The head node is selected in such way that it has the most number of neighboring nodes, as this can help our ants to traverse in most possible directions of the network. The ants in this proposed algorithm will continuously originate from the Head node. The number of ants is limited to avoid the congestion of network. The number of ants is in a limit by setting a suicide timer on each ant, which when reaches zero the ant will terminate itself. The selection of timer value would depend on the size and number of nodes in the network. These ants traverse the width and length of the network in such a way that they know about the location of underloaded or overloaded nodes in the network. These Ants along with their traversal will be updating a pheromone table, which will keep a tab on the resources utilization by each node.

A. Pheromone updation

In classical ACO algorithm, ants use pheromone trail to select the next node. This classical ACO algorithm fails to function in reverse situation. For example in Table I, node B is overloaded but now suppose that the situation is reversed i.e. node B is now underloaded. In the table I, pheromone trails of node B is strong only with D which would make the most probable next state after B as D even in this reversed situation. Therefore, proposed algorithm considered two types of pheromone, Foraging Pheromone (FP) and Trailing Pheromone (TP). There are two types of ant movement for load balancing, forward movement and backward movement.

Table I Pheromone trail classical algorithm

Node	A	B	C	D
A(O)	-	L	L	H
B(O)	L	-	L	H
C(M)	L	L	-	H
D(U)	H	H	H	-

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. 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.

In proposed algorithm, ant would lay down foraging pheromone (FP) after encountering underloaded node for searching overloaded node. Trailing pheromone (TP) is used to find path to the underloaded node after encountering overloaded node.

The formula for updating Foraging Pheromone (FP):

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

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,

∆FP = Change in the FP.

The formula for updating Trailing Pheromone (TP):

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

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,

∆TP = Change in the TP.

Following are the steps of load balancing algorithm based on ACO.

[Start]
Step 1: Initialize the pheromone tables
Step 2: Declare a threshold level for nodes
Step 3: Ants move through nodes.
Step 4: (if else condition)
if(ant-timer > counter)
stop()
Step 5: else
(if else condition)
Check the status of encountered node.
(i.e. whether encountered node
is overloaded or underloaded)
if(node is underloaded) then
follow steps from 6 to 8
else follow steps from 9 to 11
else go to step 12
Step 6: Get neighbor nodes of encountered node
Step 7: Select most loaded node from neighbor nodes
Step 8: check (if condition)
Load on selected node > Threshold
If yes then update FP and
then redistribute load and then go to
step 3. Else, again select node which
have maximum of FP among neighbor
nodes and then update FP and move to
that node. Goto step 6.
Step 9: Get neighbor nodes of encountered node
Step 10: Select least loaded node from
neighbor nodes
Step 11: check (if condition)
Load on selected node <Threshold
If yes then update TP and then
redistribute load and then go to step 3.
Else, again select node which
have minimum of TP among
neighbor nodes and then update TP
and move to that node. Goto step 9.
Step 12: (Load is equal to threshold). Select
next node randomly among neighbor
nodes. Goto step 3.

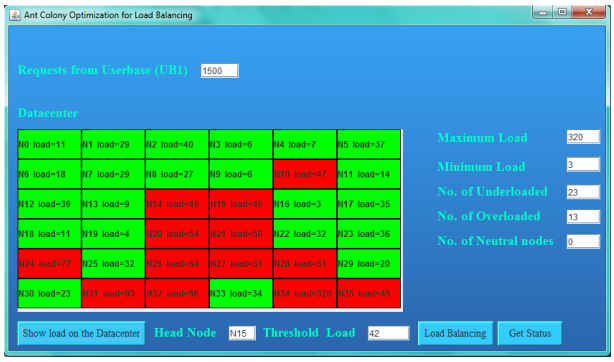

Figure I. Simulation: Load on simulated servers without Load Balancing Technique

Figure II. Simulation: Load on simulated servers with Load Balancing Technique

IV. RESULTS

Simulation is done using NetBeans. For simulation, 36 nodes are considered in Datacenter. Head node ID is 15. Threshold calculation is based on number of requests and available nodes in system. Figure I show the load on simulated servers without a load balancing technique. Number of requests from user base are 1500. Threshold is 42. Number of underloaded (green), overloaded (red) and neutral (yellow) servers are 23, 13 and 0 respectively. Now, figure II shows load on simulated servers with load balancing technique. After load balancing, number of underloaded (green), overloaded (red) and neutral (yellow) servers are 4, 0 and 32 respectively. Therefore, results shows that the proposed technique of load balancing for servers of datacenter of cloud based on ACO gives efficient resource utilization.

V. CONCLUSION

As a large number of requests are submitted to the data center, load balancing is one of the main challenges in cloud computing. 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. In this paper, result shows that proposed load balancing technique based on Ant Colony Optimization give efficient resource utilization. Future work is to implement this technique with the consideration of server's CPU power, memory etc while redistributing load.

REFERENCES

[1] Amandeep Kaur Sidhu, Supriya Kinger, "Analysis of Load Balancing Techniques in Cloud Computing", International Journal of Computers & Technology, Volume 4 No 2, March-April, 2013, ISSN 2277-3061.

[2] Dr. D.I. George Amalarethinam, and T. Kokilavani, "Load Balanced Min-Min Algorithm for Static Meta-Task Scheduling in Grid Computing", International Journal of Computer Applications (0975 – 8887) Volume 20– No.2, April 2011

[3] Isam Azawi Mohialdeen, "Comparative Study Of Scheduling Algorithms In Cloud Computing Environment", Journal of Computer Science, 9 (2): 252-263, 2013 ISSN 1549-3636 2013.

[4] 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.

[5] Manoranjan Dash, Amitav Mahapatra and Narayan Rajan Chakraboty, "Cost Effective Selection of Data Center in Cloud Environment", Special Issue of International Journal on Advanced Computer Theory and Engineering, ISSN (Print): 2319 – 2526, Volume-2, Issue-1, 2013.

[6] 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.

[7] Ekta Gupta, Vidya Deshpande, "A Load Balancing Technique for Cloud Computing", March 2014 Cyber Times International Journal of Technology and Management.