Volume 4, Issue 2, February 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Effective Scheduling Algorithm for Load balancing (SALB) using Ant Colony Optimization in Cloud Computing

Shagufta Khan
M-TECH(CSE) RKDFIST BHOPAL
(M.P.), India

Niresh Sharma
Professor (CSE) RKDFIST BHOPAL
(M.P), India

Abstract: In today's environment, the day to day business operations of organizations heavily rely on the automated processes from enterprise IT infrastructures. The cloud computing is an internet based concept which are dynamically scalable in nature and provide the virtualized resources, which are provided as service over the internet. Load balancing is one of the main challenging issues in cloud computing which is require to distribute the dynamic workload across multiple nodes to ensure that no single node is overloaded. This paper implement SALB algorithm. The main contribution of our work is to balance the entire system load while trying to maximize and minimize the different parameter (Performance, SLA violation, minimum overhead, energy issue) . Our objective is to study existing ACO's and to develop an effective load balancing algorithm using ant colony optimization.

Keyword: SALB, ACO, Load Balancing

I. Introduction:

Cloud computing is emerging as a new paradigm of large-scale distributed computing. It has moved computing and data away from desktop and portable PCs, into large data centers[6].As a part of its services it provides flexible and easy way to keep and retrieve data and files especially for making large data sets and files available for the spreading no. of users around the world[2].Figure 1.1 shows the cloud. Cloud computing provides the different types of services that are based in pay-as-go model. User can take services on cloud ranging from web application to scientific application. These services are delivered to the customer over the internet. Generally it consist of a bunch of sever and resources to different client.

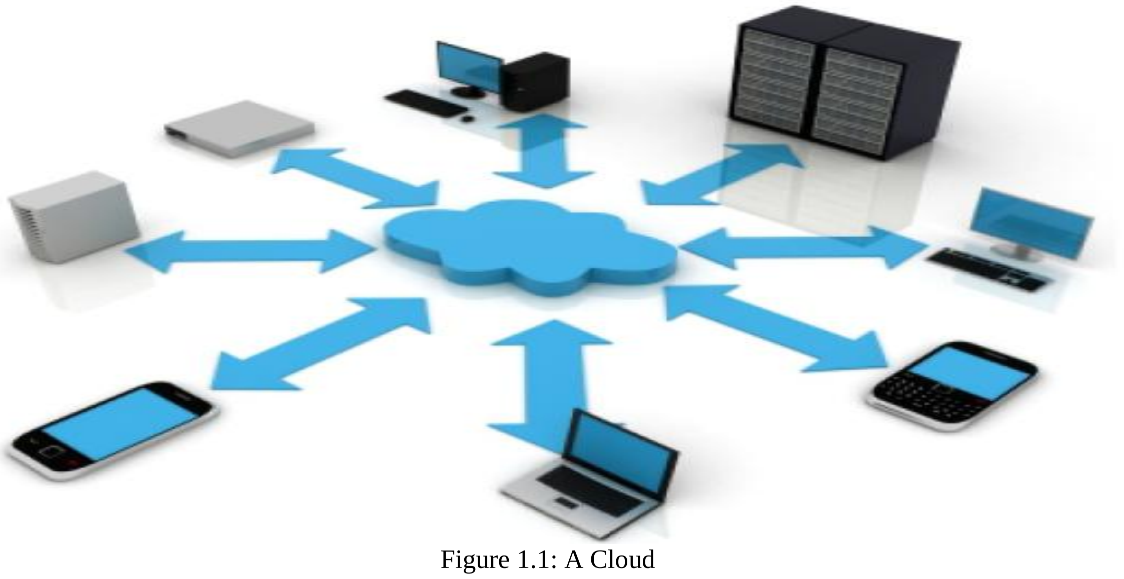

Figure 1.1: A Cloud

These services may be Saas(Software as a Services)example Salesforce[10],rackspace[7],PaaS (Platform as a Services) example Google app engine[9],window azure[12], and IaaS(Infrastructure as A Services),example .gogrid[6]. Cloud computing used the virtualization technology. This technology helps in high power server to act as multiple machines. It depends on the hardware configuration of the datacenter or server and server may contain the many number of virtual machines [16]. Load balancing is pre requirements services for increasing the cloud performance and maximum utilization of resources.It is the process of reassign the total load to individual nodes of the collective system to make the resource utilization effective and improve the response time of the job. This process removing the situation in which some of the node are overloaded while some other most under loaded [2]. This phenomenon can drastically reduce the working efficiency of the cloud.

This paper presents an ant colony algorithm, which is a bio inspired algorithm for load balancing in cloud computing environment. ACO has been previously used by several researchers in different field [1]. In this paper modified the concept of ACO(Ant colony optimization).In which movement of ant in two ways that is forward direction and backward direction and proposed the method SALB. The rest of this paper is organized as follows: Section II describes the Literature Review. Section III describes Previous Work. Section IV describes the Ant colony optimization. Section V describes the Proposed Algorithm. Section VI describes the simulation result and analysis Section VII Conclusion and Future work.

II. Literature Review

In this section we discuss the comparison of different Ant Colony Algorithm[16]. In a Cloud task Scheduling based on Load balancing ACO[3] they proposed the LBACO(Load Balancing Ant Colony Optimization) which is used to find the optimal resource allocation for each task by dynamically[16]. In this algorithm comparison with Basic ACO algorithm and FCFS algorithm, in result nodes are balanced dynamically. In addition to this, tasks are mutually independent i.e. that is there is no precedence constraints between task and computationally intensive which is not realistic for cloud system. Main drawback of this technique is the heterogeneity of system. The problem of the heterogeneity of system is removed in a technique using Load Balancing mechanism based on Ant colony & complex network theory [4] it provides the good scalability of nodes and better fault tolerant system. Practically the overhead increases during run time environment and poor response time. An Ant encounters dead state at the end due to lack of synchronization of Ants. In Cloud computing initiative using modified ACO framework [1] algorithm it minimize the make span (throughput of heterogeneous computing system).In his algorithm modification involved in basic pheromone updating formula. This modification gives the better utilization of resource but does not give the good fault tolerant system[16]. Every load balancing has primary goal the better utilization of resources. Load balancing of nodes using ACO optimization [1].It is best case for effective load balancing.

III. Previous Work

3.1 Load Balancing of Nodes in Cloud Using Ant Colony Optimization

In the previous work [1] Ant originate the from the head node. These ant traverse the width and length of the network in such a way that they know whole location of under loaded node and overloaded node. When these ants traverse the network they update the pheromone table which will store the information of utilization of each node. In the previous work movement of ant two ways: [1]

1) Forward movement-The ants continuously move in the forward direction in the cloud encountering overloaded node or under loaded node [1].

2) Backward movement-If an ant encounters an overloaded node in its movement when it has previously encountered an under loaded node then it will go backward to the under loaded node to check if the node is still under loaded or not and if it finds it still under loaded then it will redistribute the work to the under loaded node. The vice-versa is also feasible and possible [1].

The ant use two types of pheromone for its movement these are [1]-

1)Foraging Pheromone (FP)-In a typical ACO the uses foraging pheromones to explore new food sources. In our algorithm the ant would lay down foraging pheromone after encountering under loaded nodes for searching overloaded nodes. Therefore, after an ant comes up to an under loaded node it will try to find the next path through foraging pheromone [1].

2) Trailing Pheromone (TP)-In a typical ACO the ant uses trailing pheromone to discover its path back to the nest. However, in our algorithm the ants would use this to find its path to the under loaded node after encountering overloaded node. Therefore, after an ant encounters an overloaded node it will try to trace back [1].

The main aim of the two types of pheromone updation is to classify the ants according to the types of nodes they are currently searching for. The ants after originating from the head node, by default follow the Foraging pheromone, and in the process, they update the FP trails. After coming upon an overloaded node they follow the Trailing Pheromones and simultaneously update the TP trails of the path. After reaching an under loaded node of the same type they update the data structure so as to move a particular amount of data from the overloaded node to under loaded node. Ants then select a random neighbor of this node, and if they encounter an under loaded node they start following the FP to trace an overloaded node, therefore they repeat the same set of tasks repeatedly in a network to improve the network performance[1].

3.2 Problem Formulation

- Network overhead because of the larger number of ants
- Points of initiation of ants and number of ants are not clear
- Nodes status change after ant's visits to them is not taken into account
- Only availability of node is being considered, while there are other factors that should be taken into consideration
- Full replication of data.
- Reverse direction of ant in not possible.

- Overloaded node can be balanced only in forward direction reverse direction is not possible.

IV. Ant Colony Algorithm

Dorigo M. introduced the ant algorithm based on the behavior of real ants in 1996[14][15], it is a new heuristic algorithm for the solution of combinatorial optimization problems. Investigations show that: Ant has the ability of finding an optimal path from nest to food. On the way of ants moving, they lay some pheromone on the ground; while an isolated ant encounter a previously laid trail, this ant can detect it and decide with high probability to follow it. Hence, the trail is reinforced with its own pheromone. The probability of ant chooses a way is proportion to the concentration of a way's pheromone. To a way, the more ants choose, the way has denser pheromone, and the denser pheromone attracts more ants. Through this positive feedback mechanism, ant can find an optimal way finally [14][15]. 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 [14][15].

V. Proposed Algorithm

We utilize the characteristics of Ant algorithm . In contrast to other SALB algorithm inherit the basic idea of ACO . It consider the loading of different nodes.

5.1 Scheduling Algorithm for load balancing (SALB)

There are two types of movement of ant that is forward direction and backward direction (previous section described). Figure 5.1 shows the system architecture of our approaches.

- Forward direction (Forward Ant)-In this direction ant find overloaded node. Fig 5.1(a)
- Backward direction (Backward Ant)-In this direction the ant replaced the overload virtual machine with other node. Fig 5.1(b)

Our proposed Method consists of three modules which are as follows-

- Module-I Find Overloaded node.
- Module-II Select Virtual machine of overloaded node
- Module-III Placed the virtual machine with other node.

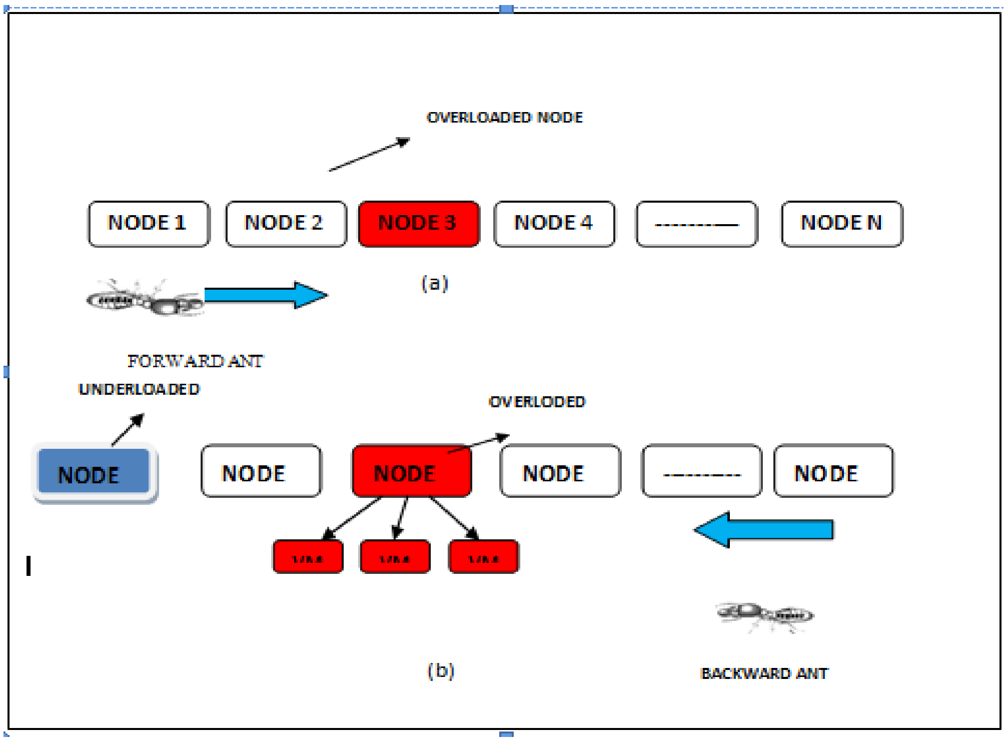

Figure 5.1:-System Architecture Of ACO

5.1.1 Module-I Find Overloaded node.

This module is based on idea of setting utilization threshold for hosts and keeping the total utilization of CPU by all the VM between these thresholds. If the CPU utilization exceeds the upper threshold, some VM from the host has being placed to reduce the CPU utilization. In this module ant find the overloaded node for doing this ant moves the forward direction. Initially ant assign the threshold value to all the node and then ant calculate the CPU utilization of each node , forward ant also maintain the pheromone table .Pheromone table contains the threshold of each node and CPU utilization of each node. If the CPU utilization is greater than the threshold then node is overloaded. Module I return the overloaded Node id to the Module II. Figure 5.2 show the flowchart of module-I.

Algorithm 5.1

```
Input-N(no. of nodes in datacenter)
Output-Find Overload node
Begin
   Initialize pheromone table
   for Node 1 To Node N
        pheromone(nodeid,threshold,cpuutilization)
         enter nodeid
         calculate cpuutilization
   end
    if cpuutilization > threshold
     then
       node is overloaded
       select nodeid
   end
end
```

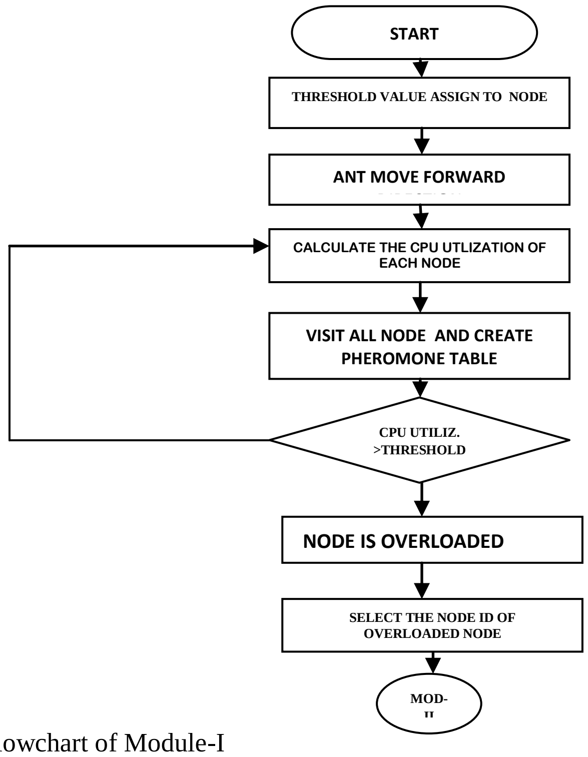
START
THRESHOLD VALUE ASSIGN TO NODE
ANT MOVE FORWARD
CALCULATE THE CPU UTLIZATION OF EACH NODE
VISIT ALL NODE AND CREATE PHEROMONE TABLE
CPU UTILIZ. >THRESHOLD
NODE IS OVERLOADED
SELECT THE NODE ID OF OVERLOADED NODE
MOD-II

Figure 5.2 : Flowchart of Module-I

5.1.2 Module-II Select Virtual machine of overloaded node.

In this module, selection of VM is based on minimum migration time. The migration time is calculated as the amount of RAM utilized by the VM divided by the network bandwidth available to the host. In this module, Ant move in backward direction i.e., Ant select the particular VM. Once it has been decided that a host is overloaded, then in the next step a particular VM is selected by an Ant to place from the overloaded host. In this module after selection of VM, the host is checked again for being overloaded. If still it considered as overloaded this policy is apply again to select the another VM to placed from the host. This process is repeated until the host is considered as not overloaded. Figure 5.3 show the flowchart of module –II.

Algorithm 5.2

```
Input-nodeid,m(no. of vm)
Output-select vm(vmid)
Begin
   Select overload nodeid
    for vm1 to m
     then
        calculate migrtation time
    end
   select vm=minimum migration time
end
```

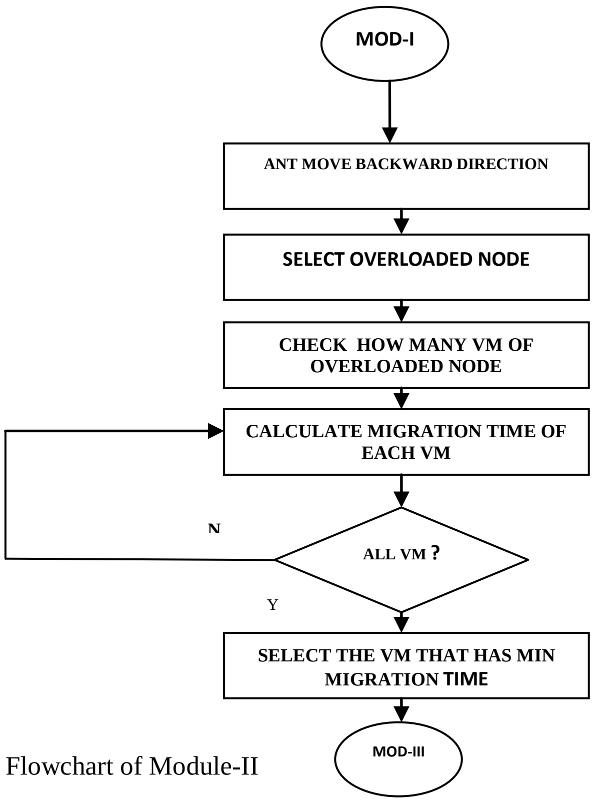
MOD-I
ANT MOVE BACKWARD DIRECTION
SELECT OVERLOADED NODE
CHECK HOW MANY VM OF OVERLOADED NODE
CALCULATE MIGRATION TIME OF EACH VM
N
ALL VM ?
Y
SELECT THE VM THAT HAS MIN MIGRATION TIME
MOD-III

Figure 5.3: Flowchart of Module-II

5.1.3 Module-III Placed virtual machine with other node.

In the third module an Ant determine the under loaded host. Firstly all the overloaded hosts are found using the method of first module and then VM will select to place from the host to another under loaded host using the method of second module, and then Ant finds the host with minimum utilization compared to other host and tries to place the VM are set to determining target node not overloaded. If this can be done, all the VM are placed to the target node and then the source host is switched to the sleep mode once all the VM placed have been done. Once it can be done then Ant update the pheromone table and entered the entry of current utilization of CPU .This process is iteratively repeated until all host that have not been considered as overloaded.

Algorithm 5.3

```
Input –virtual machine id(vmid), underload node id(unid)
Output-Balanced the node

Begin
  Select underload node from pheromone table(unid)
  redirect the cloudlet of vmid to the unid
if (node is balanced)
  then
    update pheromone table
  end
end
```

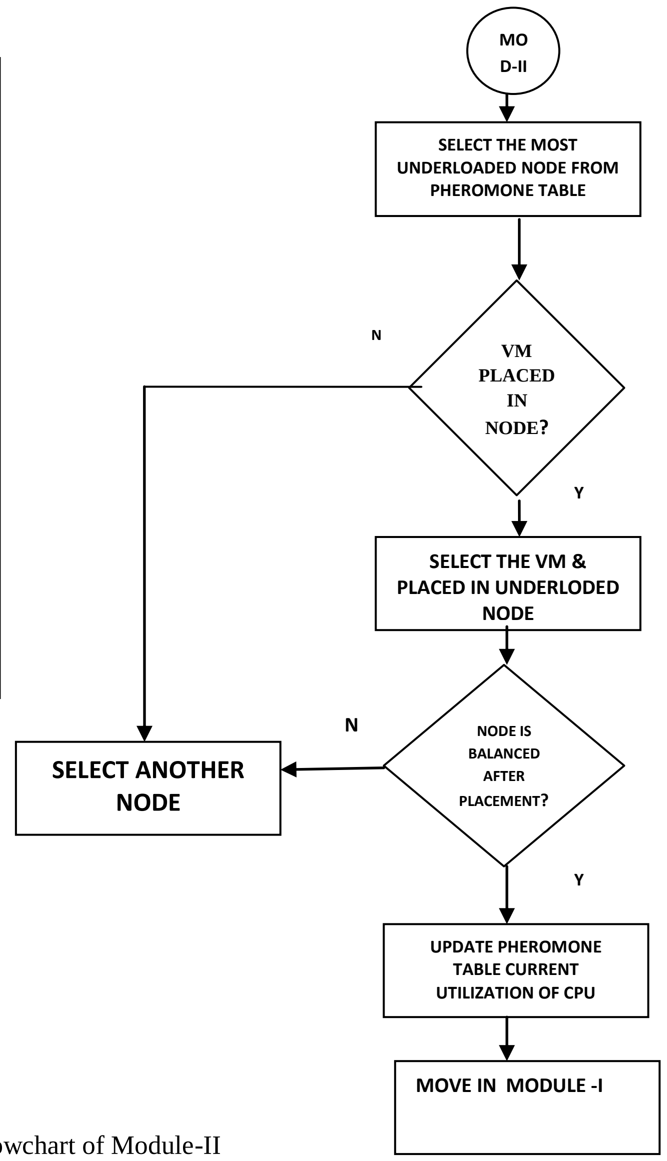
MOD-II
SELECT THE MOST UNDERLOADED NODE FROM PHEROMONE TABLE
N
VM PLACED IN NODE?
Y
SELECT THE VM & PLACED IN UNDERLODED NODE
N
SELECT ANOTHER NODE
NODE IS BALANCED AFTER PLACEMENT?
Y
UPDATE PHEROMONE TABLE CURRENT UTILIZATION OF CPU
MOVE IN MODULE -I

Figure 5.4: Flowchart of Module-II

VI. Experiment Result and simulation

6.1. Experimental Set Up

Cloud computing environments create a view of infinite computing resources to users, it is essential to evaluate the efficient resource allocation algorithm on a virtualized data center infrastructure. However it is difficult to conduct reputable large scale experiment on a real infrastructure, which is required to evaluate and compare the algorithm. To ensure the experiment, simulation has been chosen as a way to evaluate the performance of the proposed heuristics.The cloud Sim [13] toolkit has been choosing a simulation platform as a modern simulation framework aimed at cloud computing environment. The experiment is implemented on the Cloud Sim platform [13]. The scheduling algorithms of the experiment include the algorithm. We have a simulated a data center that comprises different types of physical node. The frequency of the server CPU is mapped into MIPS rating 500-2000 mips. Server is modeled to have 1 gbits per/s network bandwidth. The characteristics of VM are explained in Table 6.1 .The experiment is implemented with 1 datacenter containing two hosts and each host (host0, host 1) containing two-two virtual machine name is (vm0, vm1, vm2, vm3).In this experiment threshold value are set 80% in term of CPU utilization of host. threshold value are used for checking the status of node that node is overloaded or under loaded. If the CPU utilization will be above of 80 % then the node is overloaded. Every simulation time node is balance with under loaded node.

6.2 Simulation Result and Analysis

6.2.1 Performance metrics

Various performance metrics were taken into Consideration in order to measure the efficiency of our algorithm such s host detection and balancing the host. Energy consumption, SLA violation. These performance metrics are further explained below.

Table 6.1: parameter setting of cloud simulator

Type	Parameter	Value
Datacenter	No. of datacenter	1-10
	No. of Host	2-500
	Type of manager	Time-shared space shred
Virtual Machine	Total no. of VM	4-40
	MIPS of PE	500-2500 MIPS
	VM Memory (RAM)	512-4096 MB
	Host Bandwidth	100 Mbits/ s- 1 Gbits/s
	VM Size	2.5 GB
Cloudlets	Cloudlet	4-100
	No. Of PEs Requirements	1-4

- **Host overload detection and Balanced the Host-**

In our experiment Figure 6.1 shows that requests are go to host 1 and the simulation time when host 1 utilization is greater than the threshold then node is overloaded. Figure 6.2 shows that simulation time when host 0 and host 1 are balanced.

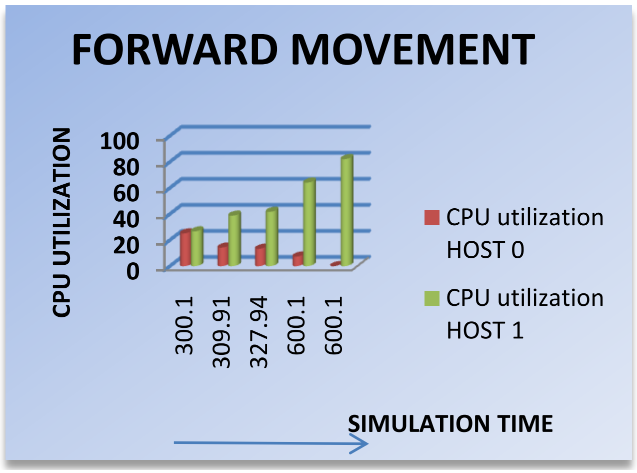

Figure 6.1: Host overload detection

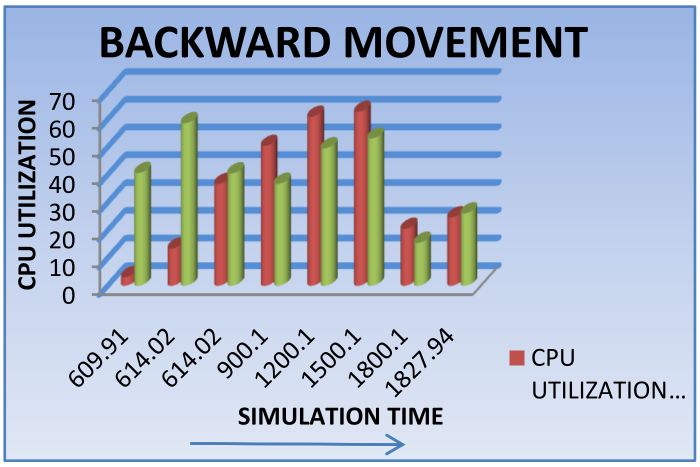

Figure 6.2 Simulation Time Of Load Balancing

- **SLA Violation**-SLA(Service Level agreement) violation metrics is very important for cloud computing environment. QOS(Quality of service) requirement are commonly formalized in the foam of SLAs.SLA representing a contract signed between customer and service provider including non functional requirement of the service specified as Quality of Service (QOS).In our experiment we define that encompasses both performance degrade due to host overloading and due to VM placement. it is denoted that when request of the host are increased SLA violation are also increased .it means that some request are violate due to host overloading . Figure 6.3 shows this experiment.

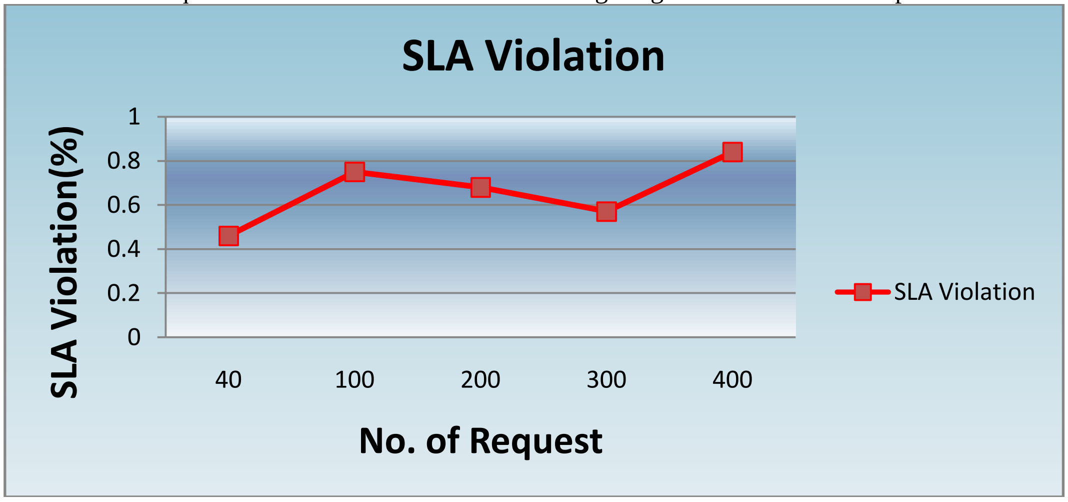

Figure 6.3 Show that SLA violation is vary with different number of requests.

- **Energy consumption** – In order to compare the efficiency of algorithm we use the several matrices to evaluate the performance. One of matrices is the total energy consumption by the physical resources of the data center by application workloads. Energy consumption is mostly determined by memory, CPU disk storage, power supplies. Fig 6.4 shows that comparison between the Ant colony algorithm and our algorithm that is SALB (scheduling algorithm for Load balancing) .In the ACO (Ant colony optimization) does not consider the energy parameter. But our algorithm gives the better performance over the ACO. It takes the less energy compare than the SALB. It means that SALB can achieve the good system balance in any situation with consider the energy issues. These result demonstrated the effectiveness of our algorithm.

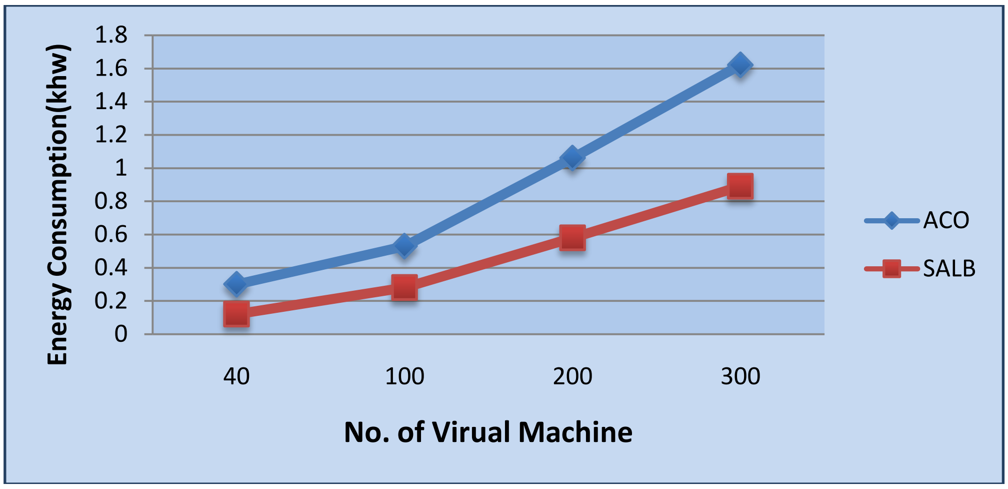

Figure 6.4: Energy Consumed by the Request

6.2.2 COMPARISION TABLE

Table 6.2: Table Under Classical Algorithm

	O-->U				**U-->O**			
NODES	**A**	**B**	**C**	**D**	**A**	**B**	**C**	**D**
A(O)		**L**	**M**	**H**		**L**	**L**	**L**
B(O)	**L**		**M**	**H**	**L**		**L**	**L**
C(M)	**L**	**L**		**H**	**M**	**M**		**L**
D(U)	**L**	**L**	**L**		**H**	**H**	**M**	

Table 6.3: Table Under Proposed Algorithm

Nodes	A(U)	A(M)	A(O)	B(U)	B(M)	B(O)	C(U)	C(M)	C(O)	D(U)	D(M)	D(O)
A(OVERLOADED)				√	√		√	√		√	√	
B(OVERLOADED)	√	√					√	√		√	√	
C(MEDIUM LOADED)	√	√		√	√					√	√	
D(UNDERLOADED)	√	√		√	√		√	√				

The Table (6.2) and (6.3) shown above describe a situation of a cloud computing with varying nodes. Their most frequent load type is denoted in the bracket (i.e. U-Under loaded node, M-medium loaded node, O-Overloaded node), in the classical table define the particular pheromone type (i.e. L-Lower, M-Medium, H-High) between the corresponding nodes . Comparison of both these algorithm is modified for balancing the node which would give the smooth functioning of cloud. In the classical algorithm two types of information are stored in the table that is Trailing pheromone and foraging pheromone, our algorithm are modified in term of forward direction and backward direction of ant. Both algorithm gives the same result but performance degradation is seen in the classical algorithm due to wastage of memory in holding the more information during the process .For example in a situation when node B is overloaded can be balanced with most probable state i.e. node D and then is C but never by A, but in modified algorithm node B can be balanced with any of the other 3 nodes. In our approaches any overloaded node can be balanced firstly with most under loaded node then medium loaded node. After finding the overloaded node an Ant can move in both direction but in classical algorithm ant cannot be move in reverse direction. This aspect is the backbone features of this modified algorithm and could not affect the efficiency of ants in the situation when maximum utilization of resources is takes place, in which the type of load on nodes choosing the opposite kind of nodes. In classical algorithm it takes more energy during the balanced process of the nodes. It can be seen in the tables and graphs attached before, that the average performance of our algorithm is better than the basic ACO and classical method of Ant colony optimization .It means that the our approaches can achieve good system load balance in any situation and take less time to execute tasks. In Other words these result demonstrated the effectiveness of our proposed algorithm.

VII. Conclusion & Future Work

In this paper an improvement version of the Ant Colony Optimization is proposed. This is a modified approach of ant colony framework that has been applied in cloud computing. Main aim of this algorithm is efficient distribution of load among the nodes. This algorithm is efficient in finding the overloaded node in minimum time. We have modified the concept of Ant colony optimization in term of movement of the ant that is in both forward direction and backward direction. The way in which ants are created pheromone table that contains the information about all nodes and its corresponding load. The goal of this study is to balance the node with efficiency & maximum utilization of resource in this ACO algorithm. A better performance is the need of our algorithm. It also improves the performance by achieving the good result in terms of throughout, response time, less energy consumption. Therefore, our proposed algorithm for load balancing in cloud Computing plays a very important role in future. There is a huge scope of improvement in this area. We have discussed on ACO algorithm that can be applied to clouds for improving the efficiency, resource utilization and energy consumption, but there is still more that can be applied to balance the load in clouds also the performance can also be increased by varying different parameters which would be our next work in future research.

Reference-

[1] R. Rastogi Kumar Nishant, Pratik Sharma,, "Load Balancing of Nodes in Cloud Using Ant Colony Optimization." Proceedings of the 14th International Conference on Computer Modelling and Simulation (UKSim), March 2012, IEEE, pp: 3-8,.

[2] Klaithem Al Nuaimi, Nader Mohamed, Mariam Al Nuaimi and Jameela Al-Jaroodi, “A Survey of Load Balancing in Cloud Computing: Challenges and Algorithms” , 2012 IEEE Second Symposium on Network Cloud Computing and Applications, 978-0-7695-4943-9/12,pp:137-142.

[3] Kun Li, Gaochao Xu, Guangyu Zhao, Yushuang Dong, Dan Wang, “Cloud Task scheduling based on Load Balancing Ant Colony Optimization”, 2011 Sixth Annual ChinaGrid Conference, 978-0-7695-4472-4/11, 2011 IEEE, pp: 3-9.

[4] Z. Zhang, and X. Zhang, “A Load Balancing Mechanism Based on Ant Colony and Complex Network Theory in Open Cloud Computing Federation”, Proceedings of 2nd International Conference on Industrial Mechatronics and Automation (ICIMA), Wuhan, China, May 2010, pp: 240-243.

[5] Amazon Elastic Computing Cloud, aws.amazon.com/ec2

[6] Cloud Hosting, CLoud Computing and Hybrid Infrastructure fromGoGrid, http://www.gogrid.com

[7] Dedicated Server, Managed Hosting, Web Hosting by Rackspace Hosting, http://www.rackspace.com

[8] FlexiScale Cloud Comp and Hosting, www.flexiscale.com

[9] Google App Engine, URL http://code.google.com/appengine

[10] Salesforce CRM, http://www.salesforce.com/platform

[11] SAP Business ByDesign, www.sap.com/sme/solutions/businessmanagement/businessbydesign/index.epx

[12] Windows Azure, www.microsoft.com/azure

[13] Rodrigo, Rajkumar Buyya “CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms” wileyonlinelibrary DOI: 10.1002/spe.995,august 2010 ,pp: 23-48.

[14] M. Dorigo, M. Birattari and T. Stutzle,” Ant Colony Optimization-Artificial Ants as a Computational Intelligence Technique” , IEEE Computational Intelligence Magazine, 1556-603X, 2006, pp. 28- 39.

[15] Jaskiran kaur Inderpal Singh,” A Survey on Ant colony Optimization ”, International Journal of Computer Science & Engineering Technology (IJCSET) ISSN : 2229-3345, IJCSET Vol. 4 No. 06 Jun 2013 ,pp:713-718.

[16] Shagufta khan ,Niresh Sharma,” Ant Colony Optimization for Effective Load Balancing In Cloud Computing, IJETTCS, ” ISSN 2278-6856, Volume 2, Issue 6, November - December 2013,pp:77-82.