ISSN No. 0976-5697

Volume 8, No. 5, May – June 2017

International Journal of Advanced Research in Computer Science

TECHNICAL NOTE

Available Online at www.ijarcs.info

Enhancement of Resource Allocation using Load Balancing in Cloud Computing

Roshni Singh
Dept. of Computer Science and Engg
MMM University of Technology
Gorakhpur, Uttar Pradesh, India

Dr. Shiva Prakash
Dept. of Computer Science and Engg
MMM University of Technology
Gorakhpur, Uttar Pradesh, India

***Abstract*:** Distributed Computing also called internet based computing is a kind of dynamic load balancing and provides on-demand for resources for services over the web. Load balancing one of the concern issues in cloud computing environment which requires distributing the dynamic workload across multiple nodes to confirm that no single node comes under overloaded or underloaded resources. VM allocation is the measure factor for efficient load balancing in Cloud Computing and allows efficient sharing of virtual machines to available datacenters. This allocation policy helps to evaluate and enhance the cloud performance. Various allocation policies are available and they have their own pros and cons. That's why we are going to overcome the limitations of existing policies by proposing a new algorithm named Hybrid Algorithm using Ant colony optimization with active clustering algorithm (HAA). This paper shows the implementation work of HAA algorithm to satisfy our objective of developing and implementing an effective load balancing algorithm.

***Keywords*:** Dynamic Load Balancing, Ant-Colony-Optimization (ACO), Active-Clustering (AC), Hybrid ACO with AC Algorithm(HAA).

I. INTRODUCTION

As cloud computing being popular the amount of processing is increases. Cloud is collaboration by various resources which perform action according to the user demand. As the demand of users can be random to resources and thus the load on each resource also vary. So every resource are unevenly loaded of tasks according to the amount of workload requested by users. And this phenomenon, reduce the working efficiency as some resources are overloaded will have a high task execution time as compared to an underloaded resources in the same cloud. This issue[12] is not only related to cloud but related to every large network. Load balancing[1][2] is the strategy of reassigning the total workload of the system to the individual points to make resource utilization convincing with efficient and to improve the response time of the task and meanwhile taking out a mode in which a couple of centers are overloaded and others center points are underloaded. In large distributed server systems it is a complex optimization problem in cloud systems and datacenters. Load balancing algorithms are classified as static and dynamic algorithms. Static algorithms[3] are suitable for homogeneous and stable environments and for this prior knowledge about the system are important. Dynamic algorithms[3][10] are more adaptable and think about various sorts of properties in the framework both preceding and amid run-time and can adjust to changes, give better outcomes in heterogeneous and dynamic conditions. Therefore some of these calculations could wind up noticeably wasteful and cause more overhead than should be expected bringing about a general debasement of the administrations execution.

Static load balancing algorithms[3] includes some algorithms as follow as :

- Round-robin Algorithm
- OLB (opportunistic Load balancing) Algorithm
- Map Reduce Algorithm
- Min-Min Algorithm
- Max-Min Algorithm

Dynamic load balancing algorithms[4] includes as follow:

- Ant Colony optimization (ACO) Algorithm
- Honey-bee Foraging Algorithm
- Biased Random Sampling Algorithm
- Active Clustering (AC) Algorithm

The various performance metrics for load balancing in clouds environment estimated are [3][4]: Throughput which evaluate the no. of jobs completed & to improve the system performance this should be high: Overhead which generate during the implementation of load balancing algorithm. It occurs due to migration of job and it should be low for good performance within the system; Fault Tolerance is the uniform nature of the load-balancing and any algorithm should be good to handle it; Response Time is rate at which particular resources executed within the distributed system and it should be minimum; Resource Utilization check the proper utilization of resources and should be optimized in nature. Scalability is performs of algorithm with finite no. of nodes. This metric should be improved; Performance metrics actually check the efficiency and should decrease the job response time while keeping transmission delays.

The main aim of this report is to survey some of existing techniques, describe their properties, and clarifies their pros and cons and then proposed an HAA algorithm for load rebalancing. The main goals are to studying the existing load balancing mechanism and providing a new classification of load balancing mechanisms. Outlining the key areas where new researches could be done to improve the load-balancing algorithms. To proposed an optimized load balancing system for cloud by using AC algorithm and Ant Colony Optimization. Using these methods, system will be less complex and time will be reduce for user request as well as for data centre request servicing time. Here the work is done on local host server method and develop a noble approach called HAA (Hybrid Algorithm with AC and ACO) for resource allocation in cloud request services on VM.

II. Related Work

Honeybee Foraging Algorithm

M. Randles et al. [5] researched the decentralized honey bee based LB procedure that is a nature-propelled calculation for self-association. It accomplishes worldwide LB through nearby server activities. Execution of the framework is improved with expanded framework assorted qualities yet throughput is not expanded with an expansion in framework measure. It is most appropriate for the conditions where the differing populace of administration sorts is required. They additionally explored an appropriated and versatile LB approach that utilizations arbitrary examining of the framework space to accomplish self-association accordingly adjusting the load over all resources of the framework. The execution of the framework is upgraded with high and comparative populace of assets hence bringing about an expanded throughput by successfully using the expanded framework assets. It is debased with an expansion in populace differing qualities. These papers additionally explored a self-collection LB method that is a self accumulation calculation to streamline work assignments by interfacing comparable administrations utilizing neighborhood re-wiring. The execution of the framework is improved with high assets in this manner expanding the throughput by utilizing these assets viably. It is debased with an expansion in framework differing qualities. This calculation depends on the conduct of the bumble bees. Two sorts of bumble bees are there-one who finds the nectar and the other who harvests. Discoverer bumble bees go looking for the nectar and discover the nourishment sources. So also in the bumble bee scavenging calculation in CC, a few servers are assembled together as a virtual server. Here the servers resemble bumble bees and the web applications resemble sustenance sources. Then the forager picks the irregular virtual server. For handling the demand, every server/resource keeps up a line. After handling, the server finds the aggregate benefit. Relies on the benefit commitment, server forms the demand. On the off chance that the ascertained benefit is less, the server comes back to their scrounge. So this keeps up the adjust of the load of the framework. However the calculation of the benefit may bring about an extra overhead which brings about general decline in the throughput.

Ant Colony Optimization

In the past work [6] ants begins the from the head hub. These ants travel entire width and length of the system in an approach to know entire area of underloaded assets and overloaded resources. At the point when these ants venture to every part of the framework then just they refresh the status table which store the data of use of every resource. The movement of ants are in two ways: Forward movement- The ants continuously move in the forward direction, finding overloaded or underloaded resources. Backward movement- If an ant finds an overloaded resources in its movement when it has previously finds an underloaded resources then it will go backward to the underloaded resources to check if it is still underloaded or not and if yes then it will redistribute the workload to the underloaded resources. The ant uses two pheromone status table for its movement as follow –

Foraging Status Table (FST)-In this, ant uses foraging pheromones table for searching to investigate new nourishment sources. So, in our method we set down rummaging status in the wake of discovering underloaded assets for seeking overloaded resources. In this way, after an insect comes up to underloaded resources it will attempt to locate the following way through scrounging status.

Trailing Status Table (TST)-In this, ant uses trailing pheromone status table to find its way back to home. Be that as it may, in our algorithm the ants utilize this to discover its way to the underloaded assets subsequent to finding over-burden asset. In this way, after a subterranean insect finds an over-burden asset it will attempt to follow back.

The objective of the two pheromone updatation status table to divide the ants as per the resources they are finding. After coming upon an overloaded resource they follow the TST and update it. In wake of coming to an underloaded asset of a similar sort they refresh the information structure in order to move a specific measure of information from the over-burden assets to underloaded. Ants then select an arbitrary neighbor of this and discover an underloaded assets they begin taking after the FST to follow an over-burden assets, along these lines they rehash a similar arrangement of undertakings more than once in framework to enhance the execution.

Modified Ant Colony Optimization

Sheeja et al.[6][7] developed a calculation in light of Ant's Behavior. Insect's Algorithm depends on the subterranean insect's conduct as the ants move towards the locale of extensive measure of assets. An insect is dependably looking for new sustenance and utilizations this nourishment sources to go back to their destination. In this calculation, initial master node resource is picked. The determination of the master node depends on the nodes which has huge number of the neighboring nodes. The subterranean insect is constantly begun from the master resource. Like an insect moves in one bearing at any given moment in look for the nourishment & in wake of getting the sustenance it goes in reverse towards its home, comparatively this calculation developed that the subterranean insect moves in forward course experience the over-burden or under stacked resource. In the event that the subterranean insect finds an over-burden resource while already it finds under stacked resource, then it will go in reverse and check if the resource is still under stacked. On the off chance that it is under stacked then the work is dispersed over the under stacked resources. So this is a productive system of asset use. In any case, the primary restriction is that because of expansive measure of ants the system might be congested. The status of the resource after the insect's visit is likewise not considered.

Optimal Virtual Machine Assign Algorithm

Shridhar G.Damanal et al.[8] developed optimal VM Assign LB Algorithm for proficient usage of VM s to appropriate and dole out the workload on the minimum stacked VM s. It upgrades the asset usage. This figuring finds the base stacked VM and dispense the work in like way. The load balancer keeps up the document of VM s. At start, all the VM s are free so they take after the round robin estimation at first however as the accompanying requesting comes, the figuring checks the VM table. In the event that the asked for VM[9] is accessible and is not yet doled out, then the asked for VM is quickly allowed to it. In the event that it is not accessible right now implies on the off chance that it is as of now doled out to do other work, then the other next minimum stacked VM is checked in the table and the workload is allocated.

Central Load Balancer algorithm

Sahu et al. [9] Developed Central Load Balancer algorithm. This algorithm distributed the workload among VM s and is based on hardware configuration and the computing capabilities. The better and reliable resource utilization is achieved through this LB algorithm. In this algorithm, first the request arrives at the Data Center. Data Center then asks the central load balancer to allocate the requests. It maintains the table that contains the VM ID, states and the set priorities of the VM s. States corresponds to the status of the VM s either busy or available. The next step is to check the priorities of the VM s so that the work is distributed accordingly. If the VM state is available, then the available VM ID is returned to the Central Load Balancer. Load Balancer is responsible for the priority computation. After the allocation of the load to the VM , the table is updated. If the VM is not available means it is busy, then the ID1 is returned and the request is queued in data center controller. Central Load Balancer is linked with the users and the VM s. It is used to calculate the priorities and then allocates the work based on the processing speed of the machine. Priority depends on the two factors including processor speed and the memory of the machine. After the VM finishes the request, Data centre gives the notification to the Central Load Balancer of the completion of the request. Now the data centre checks the next request in the queue and is available the above steps is repeated. So „Central LB‟ aims to enhance the resource utilization by avoiding the problem of under loading and the over loading. This algorithm efficiently shares the load among the VM s.

Power Aware Load Balancing

Galloway et al. [11] developed Power Aware LB (PALB) Algorithm. PALB calculation was planned with the reason to give calculation control to the group controller. PALB has three fundamental areas including adjusting segment, upscale segment and the downscale segment. Adjusting area checks the condition of the VM. The issue of irregular load in the CC is additionally improved by the usage of Equally Spread Active Execution (ESAE) calculation. In the developed algorithm as the errands are submitted, they are lined. On the off chance that the errand measure and the span of the VM coordinate, the employment is doled out by the occupation scheduler in light of the need. This calculation along these lines upgrades the reaction time. Because of equivalent spread of the employment, general cost is lessened.

Active Clustering

Active Clustering [12] works on the concept of grouping similar resources together and working on these groups. The process involved is:

- A resource initiates the process and selects another resource called the matchmaker resource from its neighbors satisfying the condition that it should be of a different type than the former one.
- The matchmaker resource then forms a connection among a neighbor of it which is of the same type as the initial resource.
- The matchmaker resource then detaches the connection between itself and the initial

Problem Statement

Most of the Load Balancing algorithms suggested is complex in terms of structure with Time Consuming. In the previous work of ant originate [4][5] from the head node. Network overhead is high because of the larger number of ants which act as nodes, points of initiation of ants(nodes) and number of nodes is not clear. Only availability of node is taken while some other are also there, which can be taken into consideration. Due to forward movement again and again problem with replication of data also exist. VM allocation is the measure factor for efficient load balancing in Cloud Computing and allows efficient sharing of virtual machines to available datacenters. This allocation policy helps to evaluate and enhance the cloud performance. Various allocation policies are available and they have their own pros and cons. That’s why we are going to overcome the limitations of existing policies by proposing a new algorithm named Hybrid Algorithm using Ant colony optimization with active clustering algorithm (HAA). The main motto of our work is to rebalance the whole system load while trying to maximize the throughput of the system and minimizing overhead and response time.

III. PROPOSED WORK

We utilize the feature of Ant Colony Optimization algorithm. Also we inherit the basic idea of ACO and AC for proposing our HAA algorithm. It considers the loading of different resources.

Hybrid Algorithm Using ACO and AC for Load Balancing (HAA)

There are two movements of ant’s direction: forward movement direction and backward movement direction

- Forward Movement (Forward Ant)-Here, ant find underloaded nodes and overloaded node.

- Backward Movement (Backward Ant)-Here, the ant replaced the overload VM with underloaded node and assigning the VM for their utilization.

Our proposed Methodology follows two phase as :

- Phase-I Find Overloaded resources and Underloaded resources.
- Phase-II Select VM for overloaded and Underloaded resources and balancing of resources.

Phase-I Find Overloaded resources and Underloaded resources

This Phase is gives idea of setting utilization of resources with size for hosts while ensuring the total utilization of resources through all the VM. In case, if the resource utilization exceeds VM size then resources are overloaded and if not then resources are Underloaded. Initially ant adds the resource size to the entire node and then evaluate resources utilization of individual node, forward ant maintain the status table as well by updating it. Status table includes the size of each node and VM utilization of each node. We apply active clustering which firstly check the matchmaker equal to VM size, if yes then redirect to status table and if not then check the VM size with resource size for finding the overloaded and Underloaded resources.

Algorithm 1(Phase I: Find Overloaded resources and under loaded resources)

```
Input: n ( no. Of resources_task)
Output: Find the overloaded resources and underloaded resources
{
Begin
        Initialize Status_table
For
        Resources_task 1 to n
        Status_table (R_id, R-size)
     Apply AC
     {
          Begin
               Matchmaker = VM_size
               Status_table (VM_id, VM_name, Password, VM_size)
          If (VM_size > R_size)
          Then
          Resource is overloaded
          Else
          Redirect to status_table
          Elseif
          Resource is underloaded
          }
     End
End
```

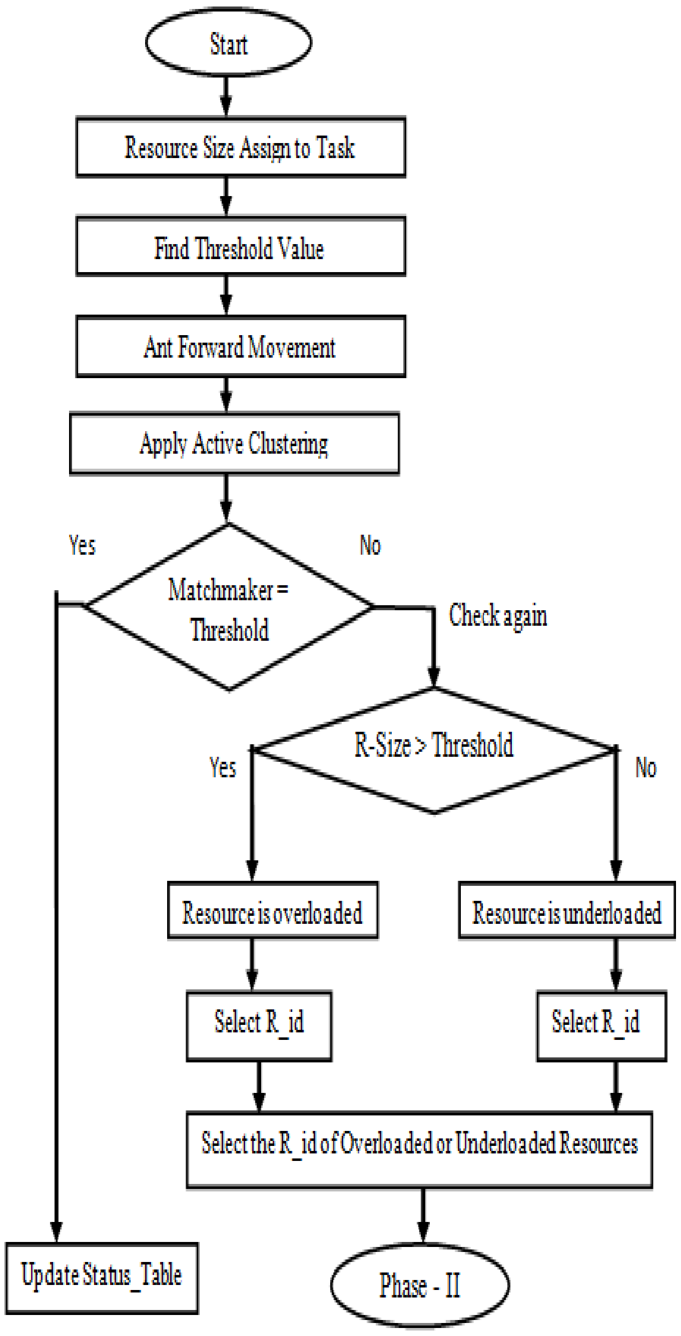

Fig 1: Flow Chart of Phase-I

Phase-II Select VM for overloaded and Underloaded Resources and Balanced the resources

In this phase, determination of VM depends on least execution time and it is computed as the measure of memory used by VM isolated by user-request for resources. Ant travel in backward way means that ant choose the specific VM. After determination of VM, the host checked again for overloaded and Underloaded resources. In the case if it is still overloaded then this arrangement is redirect again to choose another VM to put from the host. This procedure is rehashed until the host is considered as not overloveded. Now ant determine the underloaded resources. Firstly all the overloaded resources and underloaded resources are found using the method of first phase, afterward VM choose to figure normal execution time and after ant finds the least use measure nodes and tries to put the VM are set to deciding the resources not overloaded. Once it can be done then Ant update the status-table and entered the entry of current utilization of resources. This process is iteratively repeated until and unless all resources have not be considered and balanced. that have not been considered as balanced.

Algorithm 2: (Phase-II: Select Virtual machine for resources and balance the Resources)

```
Input: R_id , m ( no. Of VM)
Output : Selection of overloaded and underloaded resources
            & Balanced the resource
{
Begin
    For VM 1 to m
    Select overloaded R_id from Status Table
    Assign VM to selected Overloaded Resources

Then
    Calculate the execution time
}
    For VM 1 to m
{
    Select underloaded R_id from Status Table
    Assign VM to selected Underloaded Resources

Then
    Calculate the execution time
}

Select VM= minimum execution time
      If (Resource is balanced)
Then
     Update status table
Else
      Redirect to pervious Status Table
End
```

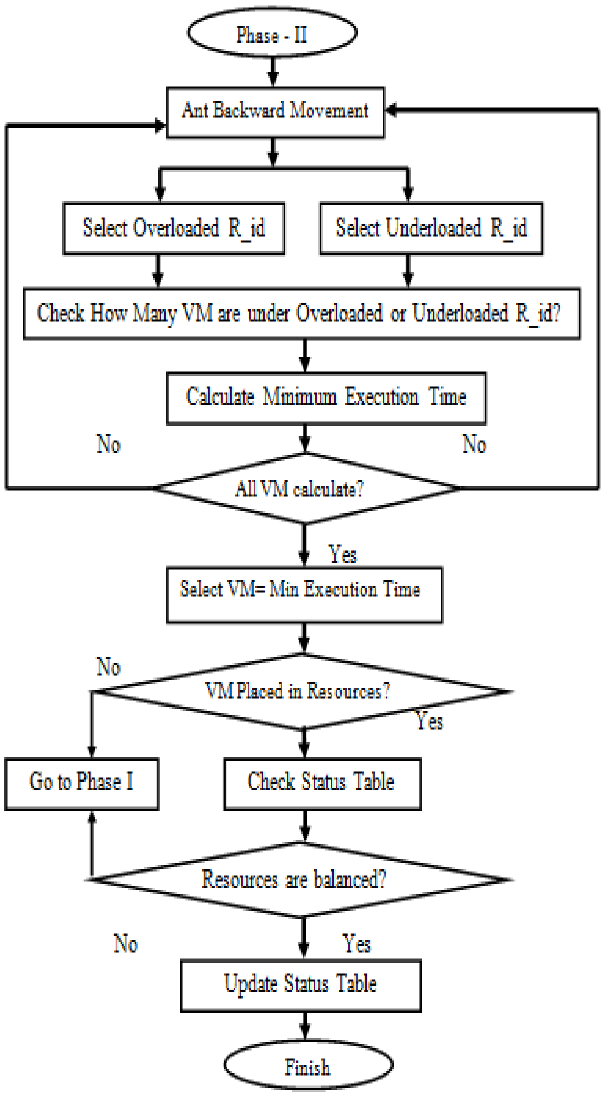

Fig 2: Flowchart for Phase II

Procedure for VM to Equally Distributed Current Execution of Load

1. Search the next available VM
2. Check all current task allocation capacity i.e threshold < max VM Size list assign the VM.
3. If available VM is not assigned.
4. Count the active current load on each VM
5. Redirect to the R_ID of those VM having underloaded resources.
6. The VM Load Balancer will assign the task request to one of the VM.
7. If VM=overloaded then VM Load Balancer will distribute some of its load to the VM having the least load in a way that every VM is equally loaded.
8. The local host server receives the resource's task send by user and then allocate to waiting requests of task queue to the available VM and so on.
9. Then again continue with 2nd step.

IV. EXPERIMENT RESULT AND IMPLEMENTATION

Experimental Set Up

CC environments make a point of view of vast resources to users because it's important to locate the effective resource provision on virtualized datacenter system. However, it's difficult to lead on respectable extensive scale test on a genuine structure which is required to discover it with comparison. To confirm the experiment, implementation is done to evaluate the overall performance of the our proposed work. The Netbean platform has been choosing for implementation as a java framework. The work is implemented on Netbean IDE platform using Tomcat Server. We have a implemented a local host server that comprises with different resources. The size of the server is designed into 5000-600MIPS. Server bandwidth is modelled to have 100-1 Gbits per seconds' n/w bandwidth. The features of VM are detailed in Table 1. This approach is implemented with 1 datacenter containing virtual machine name (vm1, vm2, vm3, vm4, vm5). In this experiment resource size are set 75% in term of VM utilization of host. Resource sizes are used for checking the status of resources i.e either resource is overloaded or underloaded. If VM utilization will be above of 75% then the resources is overloaded or less than this its goes under underloaded. Every simulation time i.e execution time of resources are balance with under loaded node.

Performance Metrics Consideration

Various performance metrics which is taken by us in consideration in behalf of finding the efficiency of HAA algorithm such as VM detection for its proper utilization and balancing node resources. We take datacenter

Table 1: Result Table

Types	Parameter Used	Value of Parameters
Datacenter	No. of datacenter	1
	No. of VM in datacenter	2-10
	Bases of Calculation	Time-Shared
Virtual Machine	No. of VM to be utilized	4
	Processing Resources in MIPS	600-5000 MIPS
	VM Capacity	
	Server Bandwidth	100-1 Gb/sec
	VM size	5 GB
Request Resources	Resources Size	200-1000 MB
	No. of Processing Resources Requirements	10

In our implementation Figure 1 shows that which VM are comes under overloaded or which comes under overloaded. If VM>threshold value then it comes under overloaded or else less then it then node is underloaded. We take VM utilization in percentage terms 75% above indicate overloaded node and less than this indicate underloaded resources.

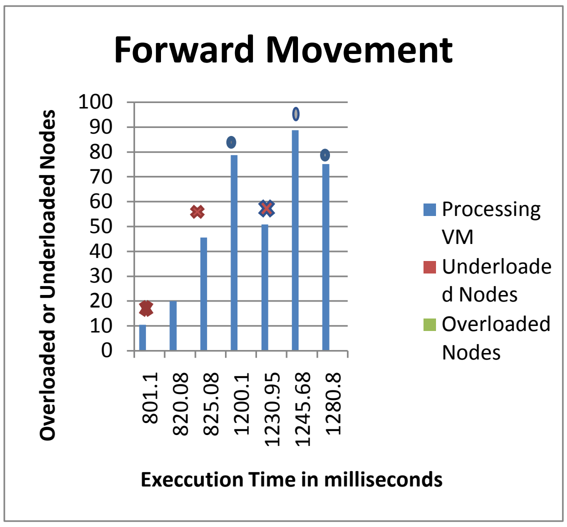

Fig 3: Overloaded and Underloaded Detection of VM

Figure 2 show that request is assigned to VM for various processing resources and we evaluated the execution time as well. This graph depicts that the VM utilization in terms of percentages. The resource demand is go to VM1 firstly and calculate its % of utilization. If VM is greater than the threshold value then node comes under overloaded or else less then it then node is underloaded. We take VM utilization in percentage terms 75% above indicate overloaded node and less than this indicate underloaded resources.

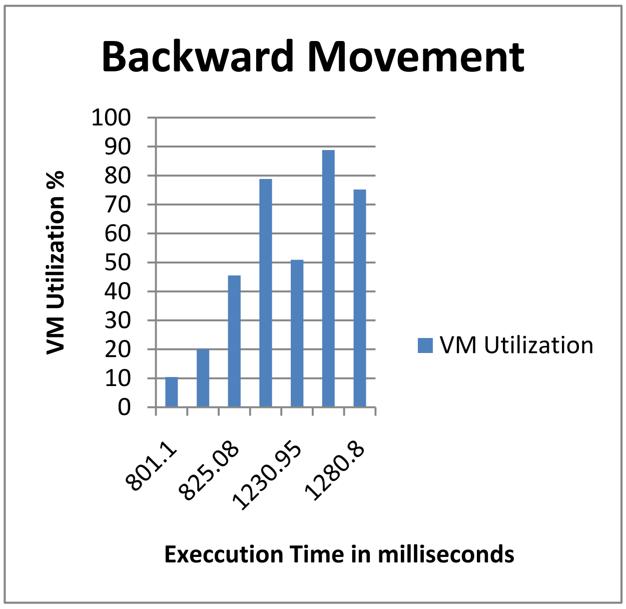

Fig 4: VM Utilization

Comparison Chart

NetBeans is used to get the LB of jobs as per the resources to be used with the help of local server host based on tom cat server. To test the efficiency of the developed algorithm, Load Balancer based on Resource Utilization, is compared with the other algorithms. The algorithm is compared with other algorithms by using cloud analyst tool. Thus, we use few of the performance metrics for comparing the efficiency of these three algorithms. One of the metric is throughput. Fig 3 gives the comparison between the pervious AC algorithms and ACO with our HAA approach. In ACO throughput is good but overhead is also maximum. So to reduce this we introduce AC within this ACO algorithm. So HAA gives better performance over ACO and takes lesser response time. It means HAA can achieve good system balancing in any situation related to overloaded or underloaded nodes. These result demonstrated the effectiveness of our approach.

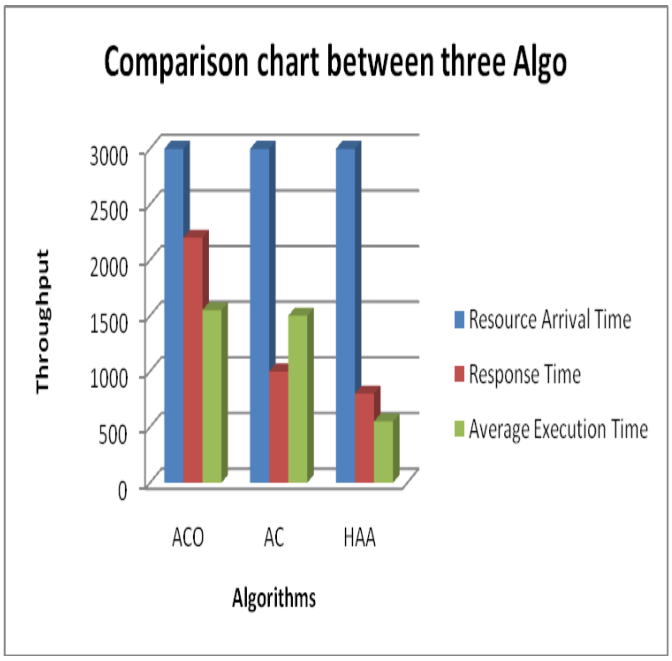

Fig 5: Comparison chart of ACO, AC and HAA

V CONCLUSIONS AND FUTURE WORK

We implemented the algorithms in Netbean IDE environment. The results show that HAA algorithm are much better than ACO algorithms and active clustering algorithm with respect to Response Time, Resource Arrival Time, Execution Time Utilizations of Tasks and assigning of VM. Although, the results of ACO are much better but by applying Active Clustering the overall performance will better. Cloud Computing is a boundless innovation and load balancing assumes a vital part in it. So there is a colossal extent of change here. I have implemented an efficient load balancing algorithm HAA by using ACO and AC yet there are as yet different methodologies that can be connected to rebalance the load within the cloud environment. The performance of our proposed algorithms can likewise be enhanced by using more parameters.

REFERENCE

[1] Aayush Agarwal, Manisha G, Raje Neha Milind,“ a survey of cloud based load balancing techniques” Department of Information Science and Engineering, P.E.S University, 100 Feet ring Road, Banashankari Stage III. Bangalore – 85

[2] Che-Lun Hung1, Hsiao-hsi Wang2 and Yu-Chen Hu2(2012), Efficient Load Balancing Algorithm for Cloud Computing Network, *IEEE* ,Vol. 9, pp: 70-78

[3] Wang S-C , Yan K-Q , Liao W-P , Wang S-S . Towards a load balancing in a three-level cloud computing network. In: Proceedings of the third IEEE international conference on computer science and information technology (ICCSIT), 1. IEEE; 2010. p. 108–13 .

[4] Jiang W , Z Jing , Li J , Hu H . A Resource scheduling strategy in cloud computing based on multi-agent genetic algorithm. TELKOMNIKA Indones J Electr Eng 2013;11(11):6563–9 .

[5] Randles M , Taleb-Bendiab A , Lamb D . Cross layer dynamics in self-organising service oriented architectures. Berlin Heidelberg: Springer; 2008 .

[6] 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.

[7] Shagufta khan ,Niresh Sharma, “Effective Scheduling Algorithm for Load balancing (SALB) using Ant Colony Optimization In Cloud Computing”, IJARCS, ” ISSN 2278-6856, Volume 4, Issue 2, February 2014,pp:77-82.

[8] 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.

[9] Wang, Shu-Ching, et al. "Towards a LB in a three-level CC network." Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on. Vol. 1. IEEE, (2010).

[10] Sahu, Yatendra, R. K. Pateriya, and Rajeev Kumar Gupta. "Cloud server optimization with Load Balancing and green computing techniques using dynamic compare and balance algorithm." Computational Intelligence and Communication Networks (CICN), 2013 5th International Conference on. IEEE, (2013).

[11] Galloway, Jeffrey, Karl Smith, and Jeffrey Carver. "An empirical study of power aware LB in local cloud architectures." Information Technology: New Generations (ITNG), 2012 Ninth International Conference on. IEEE, (2012).

[12] 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.