



# LOAD BALANCING USING THRESHOLD AND ANT COLONY OPTIMIZATION IN CLOUD COMPUTING

<sup>1</sup>Suhasini S, <sup>2</sup>Yashaswini S

<sup>1</sup>Information Science & engineering, GSSSIETW, Mysore, India

<sup>2</sup>Assistant Professor, Information Science & engineering, PESIT, Bangalore, India

Email: <sup>1</sup>Suhasini.suvi95@gmail.com, <sup>2</sup>yash.sridhar8@gmail.com

**Abstract—** Load balancing is one of the main challenges in cloud computing. It is a mechanism that distributes the dynamic local workload evenly across all the nodes in the whole cloud to avoid a situation where some nodes are heavily loaded while others are idle or doing little work. It helps to achieve a high user satisfaction and resource utilization ratio, hence improving the overall performance and resource utility of the system. It also ensures that every computing resource is distributed efficiently and fairly. Cloud computing is to utilize the computing resources (service nodes) on the network to facilitate the execution of complicated tasks that require large-scale computation. Thus, the selecting nodes for executing a task in the cloud computing must be considered, and to exploit the effectiveness of the resources, they have to be properly selected according to the properties of the task.

**Keywords—**utilization ratio; exploit;service nodes; utility

## I. INTRODUCTION

Cloud Computing is bringing application development, business, and system operations closer together. This means that software developers need to better understand business process and system operations. It also means that business stakeholders and operations staff

has to consume more software. The promise of Cloud Computing is that centralization, standardization, and automation will simplify the user experience and reduce costs.

Cloud Computing or the future of next generation computing provides its clients with a virtualized network access to applications and or services. No matter from wherever the client is accessing the service, he is automatically directed to the available resources. There are situations when our system gets hanged up or it seems to take few decades for pages to come out of printer. All this happens because there is a queue of requests waiting for their turn to access resources which are shared among them. Further these requests cannot be serviced as the resources required by each of these requests are held by another process or request by virtual machines. One cause for all these problems is called deadlock.

Load balancing is a new approach that assists networks and resources by providing a high throughput and least response time[1] . The real world example of load balancing can be a website which has thousands of users at the same time. If not balanced then the users have to face the problem of timeouts, response delays and long processing time. The solutions involve making use of duplicate servers to make the website available by balancing the network traffic.

A cloud is constituted by various nodes which perform computation according to the requests of the clients. As the requests of the clients can be random to the nodes they can vary in quantity and thus the load on each node can also vary. Therefore, every node in a cloud can be unevenly loaded of tasks according to the amount of work requested by the clients. This phenomenon can drastically reduce the working efficiency of the cloud as some nodes which are overloaded will have a higher task completion time compared to the corresponding time taken on an under loaded node in the same cloud.

## II. CLOUD COMPUTING

Cloud computing aims to share data, calculations, and services transparently among users of a massive grid. The basic principles of cloud computing is making the calculation in a large number of distributed computers, rather than the local computer or remote servers. Thus a huge computing problem can be divided into many small parts and the small parts can be distributed to many computers so that they can deal with the problems together.

Cloud computing services can be classified into three different service models:

- Infrastructure as a service (IaaS)
- Platform as a service (PaaS)
- Software as a service (SaaS)

### A. *Infrastructure as a service (IaaS)*

The first service model, which is called 'infrastructure as a service', is based on the provisioning of computing resources which are more hardware oriented. According to NIST the provisioning of "processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications", from fall under this category.

With infrastructure as a service, the user is able to run and manage own operating systems including applications by using virtualization technologies. Furthermore he can make use of storage systems and/or network devices like

e.g. firewalls. The management of the underlying infrastructure is done by the service provider of the cloud, though the user has full control over operating systems, applications and storage and eventually partial control over network devices.

With a view to mobile environments, this service model does not seem to be appropriate for mobile usage of cloud services, as it is highly focused on the provision of hardware based services with a low level of abstraction. It might only be interesting in the case of storage provisioning for mobile devices.

### B. *Platform as a service (PaaS)*

The second service model, which is called 'platform as a service', gives users the opportunity to run applications on the infrastructure offered by the service provider. However, it requires that the applications are created with programming languages or tools that are supported by the service provider. The management of the underlying infrastructure and operating systems is in the hands of the service provider. Though the user has full administrative control over the applications he wants to host on the cloud system.

Looking at mobile usage of cloud computing services, this service model seems to be of interest, because it gives users the possibility to outsource applications or parts of them to the cloud. As a result, users can make use of the benefits a cloud computing system can offer them including scalable and fast computation resources, which in the end could save time and energy.

### C. *Software as a service (SaaS)*

The third service model, named 'software as a service' focuses on the provisioning of applications. The management of the underlying infrastructure, operating systems and even the configuration of the application itself (with exemption of some partial elements) is completely done by the service provider.

With regard to mobile usage, this service model might also be of interest, although it fully depends on a working network connection

between mobile devices and the cloud system. However, the benefits of cloud systems, which might lead to savings in time and energy consumption, also apply here.

#### **D. Characteristics of Cloud computing**

- **Reduced Cost:** We want to just pay for what we use from the Cloud Computing resource pool. It does all work for us.
- **Scalability of System:** we can easily request for more processing power from the resource pool at very minimum cost according to our requirement.
- **Automatic Updates of software:** Cloud Computing Company will automatically update the software if a new version is released.
- **Remote Access of the System:** our employees and customers can access the data from anywhere around the world.
- **Disaster Relief:** The Cloud Computing Company keeps the backup of data and ensures the proper functioning of the system.
- **Quick Customer Support:** the Cloud Computing vendor provides quick customer support, which is essential for the functioning of your business.
- **Sufficient Storage:** more space is available for storage of our data.

### **III. LITERATURE SURVEY**

#### **A. LOAD BALANCING IN CLOUD**

Load Balancing is a computer networking method to distribute workload across multiple computers or a computer cluster, network links, central processing units, disk drives, or other resources, to achieve optimal resource utilization, maximize throughput, minimize response time, and avoid overload[1].

The load balancing service is usually provided by dedicated software or hardware. It further prevents bottlenecks of the system which may occur due to load imbalance. When one or more components of any service fail, load balancing helps in continuation of the

service by implementing fair-over, i.e. in provisioning and de-provisioning of instances of applications without fail. The goal of load balancing is improving the performance by balancing the load among these various resources (network links, central processing units, disk drives.) to achieve optimal resource utilization, maximum throughput, maximum response time, and avoiding overload. To distribute load on different systems, different load balancing algorithms are used.

#### **B. TYPES OF LOAD BALANCING**

There are two types of load balancing

- **Static load balancing** Static load balancing algorithms require aforementioned knowledge about the applications and resources of the system. The decision of shifting the load does not depend on the current state of the system. The performance of the virtual machines is determined at the time of job arrival. The master processor assigns the workload to other slave processors according to their performance. The assigned work is thus performed by the slave processors and the result is returned to the master processor[1]. Static load balancing algorithms are not preemptive and therefore each machine has at least one task assigned for itself. Its aims in minimizing the execution time of the task and limit communication overhead and delays. This algorithm has a drawback that the task is assigned to the processors or machines only after it is created and that task cannot be shifted during its execution to any other machine for balancing the load. The four different types of Static load balancing techniques are Round Robin algorithm, Central Manager algorithm, Threshold algorithm and randomized algorithm.
- **Dynamic Load Balancing** In this type of load balancing algorithms the current state of the system is used to make any decision for load balancing. It allows for

processes to move from an over utilized machine to an underutilized machine dynamically for faster execution. This means that it allows for process preemption which is not supported in Static load balancing approach. An important advantage of this approach is that its decision for balancing the load is based on the current state of the system which helps in improving the overall performance of the system by migrating the load dynamically[4].

### C. EXISTING LOAD BALANCING TECHNIQUE

There are different types of load balancing algorithm exists today.

- **Opportunistic Load Balancing (OLB)**

Its tries to keep node always busy it does not consider present workload on node but thing is make node always busy that means some task should assigned to all nodes. The advantage is quite simple and reach load balance but its shortcoming is not consider each expectation execution time of task, therefore the whole completion time is very poor In other words, OLB dispatches unexecuted tasks to currently available nodes at random order, regardless of the node's current workload[4].

- **Min-Min scheduling algorithm**

It is a static load balancing algorithm. So, all the information related to the job is available in advance. Min-Min algorithm begins with a set of all unassigned jobs. First of all, minimum completion time for all jobs is calculated. The job with minimum completion time is selected. Then, the node which has the minimum completion time for all jobs is selected. Finally, the selected node and the selected job are mapped. The ready time of the node is updated. This process is repeated until all the unassigned jobs are assigned. The advantage of this algorithm is that the job with the smallest execution time is executed. The drawback of this algorithm is that some jobs may experience starvation that is , it only

considers the completion time of each task at the node but does not consider the work loading of each node. Therefore, some nodes maybe always get busy but some nodes maybe still idle.

- **LBMM Scheduling algorithm**

Load Balance Min-Min (LBMM) scheduling algorithm is proposed that takes the characteristic of the Min-Min scheduling algorithm as foundation. The performance of Min-Min scheduling algorithm is considered to minimize the completion time of all works. However, the biggest weakness of Min-Min scheduling algorithm is it only considers the completion time of each task at the node but does not consider the work loading of each node. Therefore, some nodes maybe always get busy but some nodes maybe still idle. The proposed LBMM will improve the load unbalance of the Min-Min and reduce the execution time of each node effectively. In addition, the multi-level hierarchical network topology can decrease the data store cost . However, a higher lever will increase the cost of network management. Therefore, in our study, a three-lever hierarchical framework ( Figure 1.1) is used[1]. The third level is the service node that used to execute subtask. The second level is the service manager that used to divide the task into some logical independent subtasks. The first level is the request manager that used to assign the task to a suitable service manage.

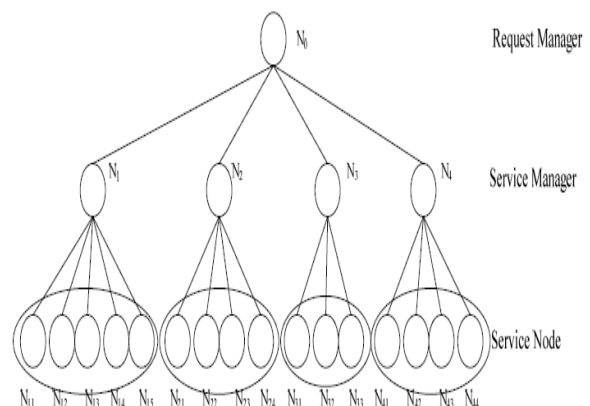


Figure 1.1 Three Level Framework

LBMM scheduling algorithm is to distribute task among each service manager into some subtasks to be executed in a suitable service node. LBMM considers the execution time of each subtask on each service node. Each subtask will be figured out the execution time on different service nodes via agent. According to the information gathering by agent, each service manager chooses the service node of shortest execution time to execute different subtasks and records it into the Min-time array. Finally, the Min-time array of each subtask is recorded that is a set of minimal execution time on certain service nodes. In the Meanwhile, service manager chooses the service node from Min-time array. This means the  $b$  subtask on service node  $a$  is performed first. Therefore, the subtask  $b$  will be distributed to service node  $a$ . Since the subtask  $b$  has been distributed to service node  $a$  to be performed, the subtask  $a$  will be deleted from subtask queue. Meanwhile, the Min time array will be rearranged, and the service node  $b$  is put on the last one of Min-time array. In our study, an integrated scheduling algorithm is provided that combines OLB and LBMM scheduling algorithm in a three-level cloud computing network[1]. According to the properties of the proposed integrated scheduling algorithm, the load balance and the execution time of each node is considered.

#### IV. LOAD BALANCING BASED ON THRESHOLD

There are several heterogeneous nodes in a cloud computing system. Namely, each node has different capability to execute task; hence, only consider the CPU remaining of the node is not enough when a node is chosen to execute a task. Therefore, how to select an efficient node to execute a task is very important in a cloud computing. Due to the task maybe has different characteristic for user to pay execution. Hence it is need some of the resources of specific, for instance, when implement organism sequence assembly, it is probable have to big requirement toward

memory remaining[4]. And in order to reach the best efficient in the execution each tasks, so we will aimed tasks property to adopt a different condition decision variable in which it is according to resource of task requirement to set decision variable. In this study, an agent mainly collects related information of each node participating in this cloud computing system, such as remaining CPU capability, remaining memory, and transmission rate. After all these data are collected, they will be provided to the manager and assist it in maintaining the load balancing of the system. The factors are defined as following:

V1 = The remaining CPU capability;

V2 = The remaining memory; and

V3 = Transmission rate

To make the manager select appropriate nodes effectively, all of the nodes (includes service manager and service node) in the system will be evaluated by the threshold that is derived from the demand for resource needed to execute the task. The service manager that passes the “threshold of service manager” considered effective, and will be the candidate of effective nodes by manager. The service nodes that pass the “threshold of service node” considered effective, and will be the candidate of effective nodes by service manager.

##### A. *Threshold of service manager*

The cloud computing environment is composed of heterogeneous nodes, where the property of each node may greatly differ. In other words, the computing capability provided by the CPU, the available size of memory, and transmission rate are different. In addition, cloud computing utilizes the resources of each node, so the available resource of each node may vary in a busy condition. From the perspective of task completion time, the available CPU capacity, the available size of memory, and transmission rate are the three decisive factors for the duration of execution. Thus, in our study, the available CPU capacity, the available

size of memory, and transmission rate were taken as the threshold for estimating service manager values. An example instance of specific as follows :

- The remaining CPU capability 490 MB/s
- The remaining memory 203698 KB/s
- Transmission Rate 7.09 MB/s

### B. Threshold of service node

When a service manager passes the “threshold of service manager” according to the requirement of a task, then the service nodes that are managed by this service manager have the capability to execute this task. However, in a cloud computing environment, the composition of nodes is dynamic, every node is likely to enter a busy state at any time and thus increase the execution time of task then lead to lower its performance. Thus, the “threshold of service node” is used choose the better service node. The progression is divided into four steps as follow:

**Step 1:** To calculate the average execution time of each subtask[1].

**Step 2:** If the required execution time of a subtask is less than or equal to the average execute time then carry out the subtask to execute normally.

**Step 3:** If the required execution time of a subtask is greater than the average execute time then the executing time is set to (the execution time is too long so that cannot to be considered). The other nodes that had been executed will reenter into the system to participate the execution of subtask.

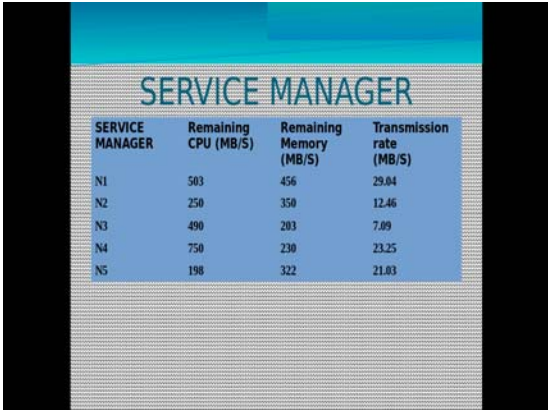
**Step 4:** Repeat Step 1 to Step 3, until all subtasks have been executed completely.

The tasks can be assigned to execute quickly by the proposed integrated scheduling algorithm and the effective service nodes can be chosen by the threshold in a three-level cloud computing network. The proposed two-phase scheduling algorithm integrates OLB and LBMM to assist in the selection for effective service nodes. First, a queue is used to store tasks that need to be carried out by manager

(N0), then the OLB scheduling algorithm within “threshold of service manager” is used to assign task to the service managers in second layer. However each consignment carries out of the task have different characteristic, so the restriction of node selection is also different. An agent is used to collect the related information of each node. According to the property of each task, the threshold value of each node is evaluated and a service node will be assigned. However, in order to avoid the execution time of some is too long and affect system performance, “threshold of service node” is used to choose the suitable service node to execute subtask.

### C. Figures and Tables

The above figure shows the available service manager along with remaining cpu time, memory and transmission rate as shown in figure 2.1



SERVICE MANAGER	Remaining CPU (MB/S)	Remaining Memory (MB/S)	Transmission rate (MB/S)
N1	503	456	29.04
N2	250	350	12.46
N3	490	203	7.09
N4	750	230	23.25
N5	198	322	21.03

Fig 2.1 figure illustrating provisioning of nodes using service manager

Consider an example consisting of four service node N11,N12,N13 and N14 and four subtasks A1,A2,A3 and A4 the average threshold is calculated and it checks the min value of subtask which can be provisioned the constraint is the sub task selected must have the value less than the average threshold

In step1 the subtask having minimum value is selected and the node provisions the service by dividing the subtask into the available nodes in our example the threshold value is 17 node A2



having value 14 is selected as shown in figure 2.2. In step2 threshold value is 18 node A1 having value 24 is selected threshold value is 24 is selected as shown in figure 2.3 .In step3 threshold value is 24.75 node A4 having value 24 is selected is selected as shown in figure 2.4.In step4 threshold value is 41 node A3 having value 26 is selected is selected as shown in figure 2.5

Even though the Min-Min scheduling algorithm, Opportunistic Load Balancing (OLB), LBMM Scheduling algorithm exist we propose a new algorithm based on threshold to get better performance by using algorithms for threshold on service nodes and service manager and ant colony optimization techniques.

SERVICE NODE	N11	N12	N13	N14	Threshold (Avg)
SUBTASK					
A1	18	14	38	26	24
A2	14	12	24	18	17
A3	26	18	66	42	41
A4	19	20	24	36	24.75

Fig 2.2: figure illustrating step1 in resource allocation from resources available in service manager

SERVICE NODE	N11	N13	N14	Threshold (Avg)
SUBTASK				
A1	18	38	26	24
A3	26	66	42	41
A4	19	24	36	24.75

Fig 2.3: figure illustrating step2 in resource allocation from resources available in service manager

SERVICE NODE	N13	N14	Threshold (Avg)
SUBTASK			
A3	66	42	41
A4	24	36	24.75

Fig 2.4: figure illustrating step3 in resource allocation from resources available in service manager

SERVICE NODE	N11	N12	N13	N14	Threshold (Avg)
SUBTASK					
A3	26	18	66	42	41

Fig 2.5: figure illustrating step4 in resource allocation from resources available in service manager

## V. ANT COLONY OPTIMIZATION

This approach aims at efficiently distribution of the load among the nodes and such that the ants never encounter a dead end for movements to nodes for building an optimum solution set. In our algorithm, first a Regional load balancing node is chosen in a CCSP, which will act as a head node. We would be referring to the RLBN[3] as head node in the rest of the paper. The selection of head node is not a permanent thing but a new head node can be elected if the previous node stops functioning properly due to some inevitable circumstances. The head node is chosen 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 of CCSP[2].

The ants in our proposed algorithm will continuously originate from the Head node.

These ants traverse the width and length of the network in such a way that they know about the location of under loaded 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. We also proposed the movement of ants in two ways similar to the classical ACO, which are as follows:

- **Forward movement** The ants continuously move in the forward direction in the cloud encountering overloaded node or under loaded node.
- **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 as shown in figure 2.A.

The main task of ants in the algorithm is to redistribute work among the nodes. The ants traverse the cloud network, selecting nodes for their next step through the classical formula given below, where the probability  $P_k$  of an ant, which is currently on node  $r$  selecting the neighboring node  $s$  for traversal, is :

$$P_k(r,s) = \frac{[\tau(r,s)]^\alpha [n(r,s)]^\beta}{\sum [\tau(r,s)]^\alpha [n(r,s)]^\beta}$$

Where,

$r$  = Current node,

$s$  = Next node,

$\tau$  = Pheromone concentration of the edge,

$n$  = The desirability of the move for the ant (if the move is from an under loaded node to overloaded node or vice-versa the move will be highly desirable),

$\beta$  = Depends upon the relevance of the pheromone concentration with the move distance.

However, with continuously originating ants at an interval of  $\Delta t$ , the overload incurred by network would increase as the number of paths followed by the ants would increase so would the cost for their maintenance and thus the network performance would take a beating. Therefore, we would keep their numbers in a limit. We can keep their numbers in a limit by setting a suicide timer on the 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. The overload would depend too much on the interval  $\Delta t$ , the smaller the overload larger the overhead and vice-versa. However, higher the number of ants more frequent would be the data changes and load balancing and thus network efficiency. For this reason, if we could limit the number of ants in the network for a good trade-off between the need to keep collecting fresh data and reduce variance, and the need to avoid congestion of the ants as well.

## CONCLUSION

The OLB scheduling algorithm is used to attempt each node keep busy, and the goal of load balance can be achieved. However, the proposed LBMM scheduling algorithm that modified from Min-Min scheduling algorithm can make the minimum execution time of each task on cloud computing environment.

The goal of this study is to reach load balancing by OLB scheduling algorithm, which makes every node in working state. Besides, in

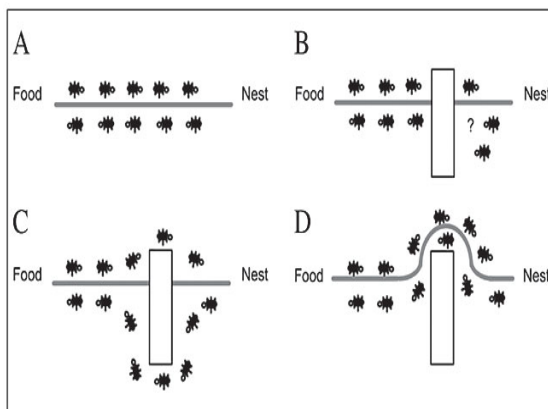


Figure 2. A. Ants in a pheromone trail between nest and food; B. an obstacle interrupts the trail; C. ants find two paths to go around the obstacle; D. a new pheromone trail is formed along the shorter path.



our research, the LBMM[1] scheduling algorithm is also utilized to make the minimum execution time on the node of each task and the minimum whole completion time is obtained. However, the load balancing of three-level cloud computing network is utilized, all calculating result could be integrated first by the second level node before sending back to the management. Thus, the goal of loading balance and better resources manipulation could be achieved.

Ant colony optimization[3] that has been applied from the perspective of cloud or grid network systems with the main aim of load balancing of nodes. The main benefit of this approach lies in its detections of overloaded and under loaded nodes and thereby performing operations based on the identified nodes. This simplistic approach elegantly performs our task of identification of nodes by the ants and tracing its path consequently in search of different types of nodes. We have used the same concepts of Ant colony optimizations and have only modified the concepts where forward and trailing pheromones are used according to our convenience.

## REFERENCES

- [1] Wang S., Yan K., Liao W. and Wang S, "towards a load balancing in a three level cloud computing network"3rdInternational Conference on Computer Science and Information Technology, 108-113, 2010.
- [2] Kumar Nishant, Pratik Sharma, Vishal Krishna,Load and Nitin Balancing of Nodes in Cloud Using Ant Colony Optimization 2012 14th International Conference on Modeling and Simulation UK.
- [3] S. Banerjee, I. Mukherjee and P.K. Mahanti, Cloud Computing Initiative using Modified Ant Colony Framework, World Academy of Science and Technology, 56, pp. 221-224, 2009.
- [4] T. Kokilavani, Dr. D. I. George Amalarethnam "Load Balanced Min-Min Algorithm for Static Meta Task Scheduling in Grid computing" International Journal of Computer Applications Vol-20 No.2, 2011.