IJCTA, 9(22), 2016, pp. 151-161
© International Science Press

Efficient Load Balancing Based on Improved Honey Bee Method in Cloud Computing

Lekha Nema* Avinash Sharma**

Abstract : The next-generation of cloud computing will bloom on how efficiently the infrastructures are instantiated and existing resources are employed dynamically. Efficient load balancing is one of the key challenges in Cloud computing. Many cloud researchers had proposed different load balancing methods for cloud computing. Existing load balancing techniques are encounters with various challenges during dynamic work load distribution for a heterogeneous public cloud in peak time. It can degrade system performance. Honey bee methods are widely used for efficient load balancing. This technique can be improved by adding some new parameters such as load calculator, priority checker, balancer status and symmetric balancing in existing method. In this paper, we are proposing a modified version of honey bee behavior inspired load balancing (EHBB-LB) algorithm, which intend to achieve well balanced load across the various virtual machines for maximizing the throughput. The planned algorithm also takes care of prioritized tasks in such a means that the quantity of waiting time of the tasks in the queue obtains minimized. Proposed EHBB-LB technique and existing load balancing methods such as FCFS and Honey bee are implemented over cloud-sim simulator 3.1 and various performance comparison parameters are calculated for instance Make-span time and execution time of tasks before and after applying load balancing. Simulation results clearly shows that proposed EHBB-LB method performs excellent in terms of makespan time and execution time as compared to existing methods.

Keywords : Load balancing, Virtual Machine, Honey bee, Cloud Computing, Cloudsim.

1. INTRODUCTION

The Cloud computing Cloud computing is a mock-up for enabling suitable, on demand network access toward a shared pool of configurable computing resources that may be very quickly provisioned and released with a minimum management effort.

1.1. Cloud Components

A Cloud system consists of 3 main components mentioned in below given figure :

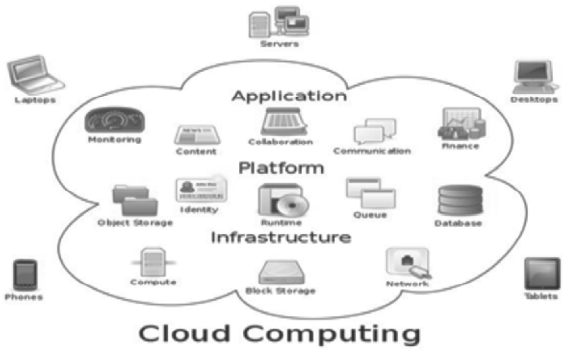

Fig. 1. Cloud Computing Structure.

* M.Tech Scholar Department Of Computer Science Oriental college of technology Bhopal, India *lekhagupta1507@gmail.com*
** Assistant Professor Department Of Computer Science Oriental college of technology Bhopal, India avisharma10@gmail.com

- **Clients :** End users cooperates with the clients to manage information related to the cloud. Clients .
- **Data Center :** Datacenter is nothing but a group of servers hosting diverse applications. An end user unites to the datacenter to subscribe different applications.
- **Distributed Servers :** Distributed servers are the piece of a cloud which is present throughout the internet hosting different applications.

1.2. Load Balancing in Cloud Computing

Load Balancing is a method in which the workload on the resources of a node is shifts to individual resources on the other node in a network with no disturbing the running task.

Load balancing in cloud using :

1.2.1. Types of Load balancing algorithms

- **Sender Initiated :** If the load balancing algorithm is begins by the sender.
- **Receiver Initiated :** If the load balancing algorithm is begins by the receiver.
- **Symmetric :** It is the combination of both algorithms sender begins and receiver initiated.

Usually depending on the present state of the system, load balancing algorithms can be alienated into 2 kinds as given in :

- **Static :** It doesn't depend on the present state of the system. Previous knowledge of the system is required.
- **Dynamic :** Results on load balancing are based on present state of the system. No previous knowledge is required. So it is superior to static approach.

  In a complex and big systems, there is a great need for load balancing. For simplifying load balancing worldwide (like in a cloud), one thing can be done is, applying the techniques would act at the components of the clouds in such a way that the load of the entire cloud is balanced. For this reason, we can use three types of solutions which can be applied to a distributed system (which require dynamic algorithm): honeybee algorithm, a biased random sampling on a random walk procedure and Active Clustering.

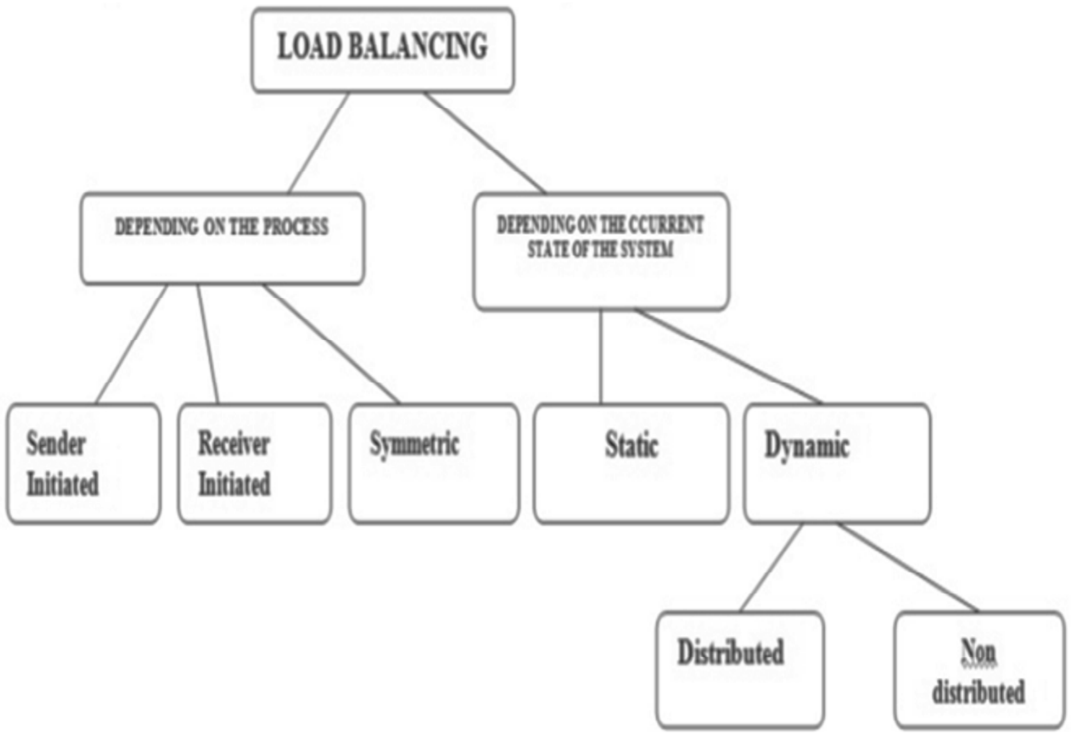

Fig. 2. Load Balancing Algorithms Hierarchy.

1.2.2. Challenges in load balancing

While developing such efficient Load balancing algorithms following challenges are come across :

- Estimation of load-Estimates the total loads in the environment.
- Comparison of load-Compare all the load to effective utilization.
- Stability of different system-Different systems have different stability
- Performance of system-How system is performing in different conditions.
- Interactions between the nodes- An important factor that how node are communication to each other.
- Nature of job to be transferred-Which type of job should be transfer.
- Selection of nodes-This load considered can be in terms of CPU load, amount of memory used, delay or Network load.

2. OVERVIEW OF SOME CLOUD LOAD BALANCING ALGORITHMS

- **First Come First Serve (FCFS) :** FCFS intended for parallel processing & is intend at the resource with the least waiting queue time and is selected for the incoming task. The Cloud-Sim toolkit maintains FCFS scheduling scheme for internal scheduling of jobs. Allocation of application exact VMs to Hosts in a Cloud based data centre is the liability of the VM provisioned components. The default policy implemented by the VM provisioned is a straight forward policy that allocates a VM to the Host in FCFS basis. The drawback of FCFS is that it is non preemptive. The shortest tasks which are at the rear of the queue have to wait for the lengthy task at the front to end. Its turnaround and response is quite low.
- **Honey Bee Colony optimization :** A lot of techniques like honeybee Foraging system, had been took place in the area of research. These techniques too motivated by swarm intelligence. This algorithm is derived from the behaviour of honey bees for finding and reaping food.

  There is a category of bees called the forager bees which forage for food sources, upon finding one; they come back to the beehive to advertise this using a dance named waggle dance. The exhibit of this dance, gives the idea of the class or amount of food and too its distance from the beehive. Scout bees then follow the foragers to the location of food and then began to reap it. They then go back to the beehive and do a waggle dance, which gives a thought of how much food, is left and hence results in further exploitation or abandonment of the food source.

  In case of load balancing, as the web servers insist increases or decreases, the services are assigned dynamically to control the changing demands of the end user. The servers are grouped under virtual servers (VS), every VS having its own virtual service queues. Every server processing a call for from its queue calculates a profit or reward, which is analogous to the worth that the bees demonstrate in their waggle dance. One measure of this reward can be the amount of time that the CPU spends on the processing of a request. The dance floor in case of honey bees is analogous to an advert board here. This board is also inured to advertise the profit of the entire colony. Each one of the servers takes the role of either a forager or a scout. The server later than processing a request can post their profit on the advert boards with a probability.

3. REVIEW OF LITERATURE

Here we are giving studies of current researches of load balancing based on cloud computing.

3.1. A cost efficient load balancing based on honey bee behaviour in VM environment

As per the fine research of authors Sheeja Y S and Jayalekshmi S [1] in cloud computing atmosphere, the load balancing of non pre-emptive autonomous tasks is a significant aspect of task scheduling. The tasks are executed on VMs and these VMs are run in parallel in order that the load has to be fine balanced across every VMs. As the cloud user pay for the service they may interested in dropping the execution time of task with the cost of using VM instances like CPU, Memory, Bandwidth etc. Projected algorithm balance the load across VMs efficiently by mapping tasks to below loaded based on the foraging behavior of honey bees and if there is more than one below loaded VMs, it too selects cost effective one via Pareto dominance approach. The experimental outcome demonstrated that the algorithm is efficient when compared with current algorithms and it too decreases the cost of using VM instances.

3.2. Best possible load balancing in cloud computing by proficient use of VMs

Shridhar G.Damanal and G. Ram Mahana Reddy [2] proposed a new algorithm that distributes the load on all existing VMs without below/above utilization. Proposed algorithm handles the load at the server by bearing in mind the existing status of the all accessible VMs for assigning the arriving requests intelligently. The VM assign load balancer mostly points on the best utilization of the VMs. They proved that proposed algorithm distributes the load and hence below/above utilization. When compared to current Active-VM load balance algorithm, the load was not correctly distributed on the VMs and from Table 3 we can say that VM 0 is used almost 10 times more than the VM 24. It shows initial VMs are over utilized and later VMs are underutilized. Proposed algorithm resolve the difficulty of inefficient utilization of the VMs or resources compared to existing algorithm.

3.3. A new honey bee enthused algorithm for dynamic load balancing in cloud environment

Chandrakanta Korat and Piyush Gohel [3] proposed an algorithm to maximize the throughput. Through this paper, they have discussed different honey bee inspired algorithms. Their proposed algorithm uses the idea of pareto dominance’s weighted sum method for selecting optimal VM and too work with preemptive task scheduling. Proposed algorithm tag on the foraging behaviour of honey bees for assigning VMs to the tasks and utilize preemptive scheduling. Pareto dominance idea is used for mutually selecting optimal VM and to locale priorities to the tasks and here numerous QOS parameters are measured. The proposed algorithm works better with numerous qos parameters. They consider fault rate of VM. So the system become more faults tolerant by using this algorithm. Here VMs are considered with processing power of 1000-7000MIPS. Weights are considered as 0.5, 0.3, and 0.2 for w1, w2 and w3 respectively.

3.4. Honey bee behavior enthused load balancing of tasks in cloud computing environments

As per research of Dhinesh Babu L.D. and P. Venkata Krishna [4] Scheduling of jobs in cloud computing is an optimization difficulty. Load balancing of non-preemptive self-governing tasks on VMs is a significant aspect of job scheduling in clouds. When certain VMs are overfull and left over VMs are under loaded with jobs for processing, the load has to be balanced to attain best machine deployment. Through this paper, authors proposed an algorithm named HBB-LB (honey bee behavior inspired load balancing), which intend to attain well balanced load across VMs for make the most of the throughput. Proposed algorithm too balances the priorities of tasks on the machines in such a manner that the amount of waiting time of the jobs in the queue is minimal. Their results demonstrate that the algorithm is efficient when evaluate with presented algorithms. Authors proposed method illustrates that there is an important development in average execution time and reduction in waiting time of jobs on queue.

4. PROBLEM DEFINATION

Existing load balancing techniques have following challenges based on survey :

- **Slower Response Time :** Slower response time leads to poor performance for the system.
- **Higher Execution Time :** Higher execution time demonstrates poor performance.
- **Selection of Load balancing :** Dynamic Load balancing demonstrates better performance.
- **Selection of partitioning method :** Many load balancing techniques are based on static partitioning, which are less efficient in huge environment.

- **Prediction of Task arrival patterns :** Jobs are arrive from various nodes in cloud environment, so it is quite hard to identified exact arrival pattern.
- **Priority of Task :** Throughout load balancing it is also challenging to execute jobs priority wise.

5. OBJECTIVE OF THE WORK

The main purpose of the work is to achieved followings :

- **Higher Response Time :** Higher response time demonstrates better performance for the system. So it is always a desirable, for better system performance.
- **Lesser Execution Time :** Lesser execution time demonstrates better performance.
- **Load balancing method type :** Dynamic Load balancing demonstrates better performance.
- **Partitioning method :** Dynamic partitioning method is required for large environment.
- **Prediction of Task arrival patterns :** Predict cloud load previous by using symmetric load balancing.
- **Priority of Task :** During load balancing it is too challenging to execute jobs priority wise.

6. PROPOSED METHODOLOGY

The proposed methodology EHBB-LB is based on proficient load balancing. It uses following key concepts :

- Load distribution are based on instructions instead of task
- **Calculate load earlier :** The basic idea behind the EHBB-LB is to recognize idle machines and resources earlier in shortest access time.
- **Dynamic load partitioning :** Proposed method is based on dynamic load partitioning for instructions.
- Random sampling redesigned the queue size, as per the load.

When a job turns up at the public cloud, the first step is to decide the correct partition. The cloud partition status can be alienated into three types :

- **Idle :** While the percentage of inactive nodes exceeds, change to idle status.
- **Normal :** While the percentage of the usual nodes exceeds, change to normal load status.
- **Overload :** While the percentage of the overloaded nodes exceeds, change to overloaded status.

6.1. Working of Proposed EHBB-LB Method

Following steps are employing in our proposed method :

- The removed tasks from overloaded VMs are measured as honey bee.
- Upon submission to the under loaded VM, the task will update the number of different priority tasks and load of that exact VM to all other waiting tasks. This will be helpful for other tasks in choosing their virtual machine based on load and priorities.
- Every time a high priority task has to be submitted to other VMs, it must consider the VM that has least amount number of high priority tasks so that the particular task will be executed at the earliest.
- Since all VMs will be sorted in ascending order based on load, the task removed will be submitted to under loaded VM.
- In essence, the tasks are the honey bees and the VMs are the food sources. Loading of a task to a VM is related to a honey bee foraging a food source (a flower or a patch of flowers).
- When a VM is overloaded *i.e.*, related to the honey getting exhausted at a food source, the task will be scheduled to an under loaded VM similar to a foraging bee finding a new food source.
- This removed task updates the remaining tasks about the VM status similar to the waggle dance performed by the honey bees to inform other honey bees in the bee hive.

- This task will update the status of the Virtual Machine i.e., how many tasks are being processed by the VM and about the number and particulars of high priority tasks currently processed by the VM in a manner similar to the bees finding an abundant food source updating the other bees in the bee hive throughout its waggle dance.
- This updating will provide a clear idea in deciding which task should be assigned to which VM based on the accessibility and load of the VMs similar to which honey bees should visit which food source rooted in whether honey is available at a flower patch or not.

6.2. Proposed Algorithm EHBB-LB for Load balancing

The proposed methodology "Enhanced HBBLB" Technique uses following steps :

Inputs : Number of Cloud brokers, server, Cloudlets, virtual machines, data centers and number of cloud resources

Output : Load balance will balance more proficiently on cloud machines and generates improved throughput

Here Task is considered as a honey bee and low loaded VMs are considered as the goal of the honey bees.

Step 1 : Cloud Basic environment variable are created

1.1. Number of Virtual Machines set Vmi (where $i = 1$ to n)

1.2..Number of cloudlets or user set CLi (where $i = 1$ to n)

1.3. Create cloud Broker B

1.4. Create data center DCi

1.5. Task represents by the set T= {T1, T2 ………… Tn}

Step 2 : Calculate cloud system capacity

2.1. Assign priority for task parameter-T_{high}, T_{med}, and T_{low} represents high, medium and low priority tasks respectively.

2.2. Assign instruction of task T_{high}, T_{med}, and T_{low} to I_{high}, I_{medium}, and I_{low} , respectively to a VM.

2.3. Calculate Capacity Cvm of a virtual machine VM.

Cvm = (Number of processors in VM) * (Number of instructions of all processors) * (communication bandwidth ability)

2.4. Calculate Capacity C of all VMs or Capacity of data center

$$C = \sum_{i=1}^{m} Ci$$

Step 3: Compute the load degree and calculate the average cloud partition degree from the node load degree

3.1. Load degree $(N) = \sum_{i=1}^{m} \alpha i \text{ Fi},$

Where αiFi, ($\sum \alpha i = 1$), shows weights that represents different values for or different kinds of jobs and N represents the current node.

3.2. Calculate the average cloud partition degree from the node load degree.

$$\text{Load\_degree}_{avg} = \sum_{i=1}^{n} \text{ Load degree (Ni)} / n$$

Step 4 : Calculate processing time of a Virtual machine and all Virtual machines.

4.1. Processing time PTi, of all Virtual machine

PTi = Load of all VMs in a data center/Capacity of all VMs

Step 5 : End User request are assign in to queue

5.1. Number of users from various location send request to cloud for their job processing Jobs Ji.

5.2. Each jobs having number of instructions or requests

5.2.1. Calculate computational time (Make span) of each job

$$\text{Makespan} = \max\{\text{CTij} \mid \text{I} €\text{T}, i = 1, 2, \ldots n$$

$$\text{Where Ct} = \text{finishing time}$$

5.2.2. Check priority for each job

5.2.2.1. If any job has higher priority in queue will take first positing in queue

Step 6: Check nodes load status levels

6.1. Idle When- Load_degree (N) = 0, There is no job being processed by this node so the status is charged to Idle.

6.2. Normal for-0 < Load_degree (N) <= Load_degree$_{\text{high}}$

The node is normal and it can process other jobs.

6.3. Overloaded When- Load_degree$_{\text{high}}$ <= Load_degree

(N) The node is not available and cannot receive jobs until it returns to the normal.

Step 7 : Proceed the jobs which are in ready queue

7.1. Submit the list of tasks T = T1, T2…Tn by the user.

7.2. Get the available virtual resources from data center.(for *i.e.*, VM1, VM2…VMm.)

7.3. Check if (Standard deviation< Threshold time) System load is balanced and Exit

$$\text{Where } \Omega = [(1/3^*)\sum_{i=1}^{m}(\text{PTi} - \text{PT})^2]^{1/2}$$

$$m : \text{No of Virtual machine Vm}$$
$$\Omega : \text{Standard deviation}$$
$$\text{PT} = \text{Load /Capacity}$$
$$\text{Ts} = \text{Threshold value}$$

7.4. Processing of queue Jobs

7.4.1. Select each job one by one

7.4.2. proceed it into the VMs by considering the assign priority and load

7.5. Compute the fitness value, $\Omega \le \text{Ts}$,

Where threshold value Ts is in between 0 and 1

7.6. Based on Fitness value, employed bees- Update the available source position by.

$$\text{Eij} = (\text{Xij} * \text{Wij}) + 2^*(\text{Lij} - 0.5) * (\text{Xij} - \text{Xkj})\text{L1} + \text{Qij}(\text{Xij} - \text{Xkj})\ \text{L2}$$

Where- $\text{Wij} = \text{L1} = 1 /(1 + \exp^*(-\text{Fitness}(i) / \text{ap}))$

$\text{L2} = 1$ if bee is onlooker one

L1, L2 – are fixed number, o or 1 $\text{L2} = 1 /(1+ \exp^*(- \text{Fitness}(i) / \text{ap}))$, if a bee is employed one

Where-

$$\text{Xij} = \text{Nearest neighborhood employed bees.}$$
$$\text{Wij} = \text{initial weight}$$
$$\text{Xkj} = \text{nearest search solution of onlooker bees.}$$
$$\text{Ap} = \text{Fitness value in first iteration}$$
$$\text{Lij} = \text{Random numbers between } [0,1] \text{ for employed bees.}$$
$$\text{Qij} = \text{Random numbers between } [0, 1] \text{ for onlooker bees.}$$

7.7. Employed bee share his information related to neighborhood position. With onlookers bee and scout bees.

Step 8. Repeat step 2 to 7, for each iterations unless a best solution is not found

Step 9. After allocating all tasks, check the load of the Vms.

9.1. If any virtual machine V_M is overloaded, it goes for next under loaded V_M and assigns the task.

9.2. After completing the each task, repeat the process for all the available jobs/tasks till the system become balanced

Step 10. Stop

7. SIMULATION AND RESULTS

We are using Cloud Sim 3.1 and its supporting tools(Java Net beans IDE) to simulate our proposed work.

Following performance comparison parameters are calculated for FCFS, Honeybee and proposed EHBB-LB method :

7.1. Makespan -can be defined as the overall task completion time

Number of task (1 task = 100 Instructions	*FCFS*	*HBBLB*	*E-HBBLB*
35	564	525	475
40	664	630	581
45	927	790	715

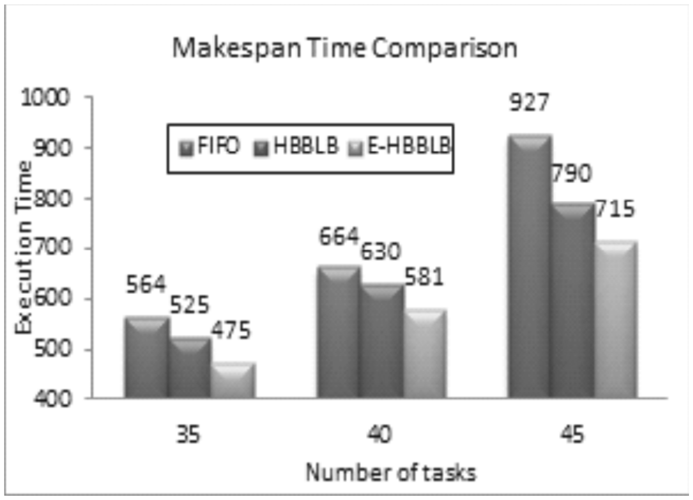

Fig. 3. Makespan time with FCFS, HBBLB & EHBB-LB

7.2. Comparison of execution time of tasks before and after applying the load balancing for proposed EHBB-LB

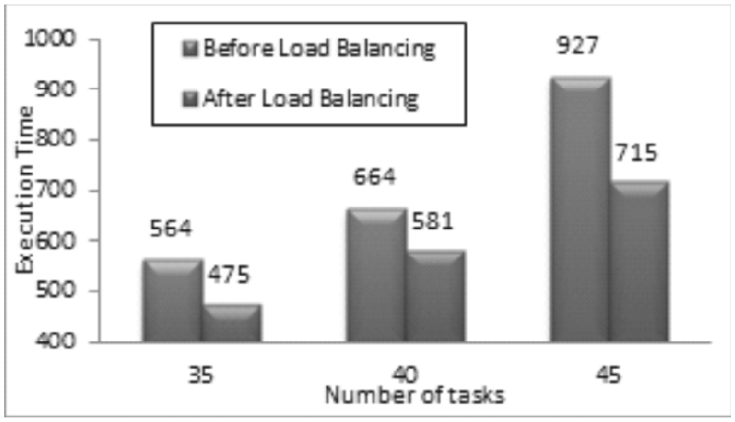

Fig. 4. Execution time of tasks before and after applying load balancing with E-HBBLB

7.3 Percentage of overloaded and non-overloaded VMs after applying HBBLB and E-HBBLB

VM status in % after Load Balancing			
HBBLB		***EHBB-LB***	
Non-Overloaded	Overloaded	Non-Overloaded	Overloaded
8020	100	0	

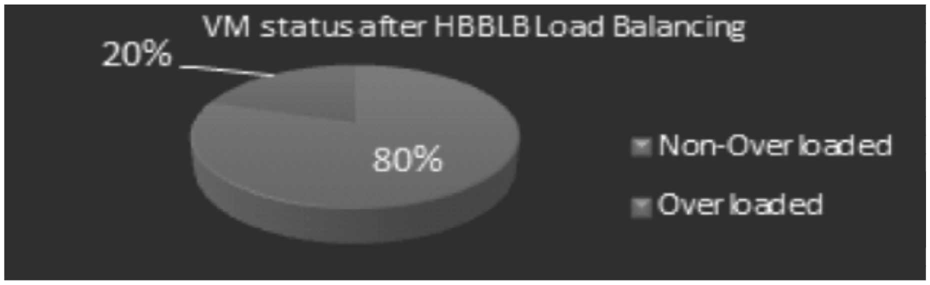

Fig. 5. Percentage of overloaded and non-overloaded VMs after applying HBBLB.

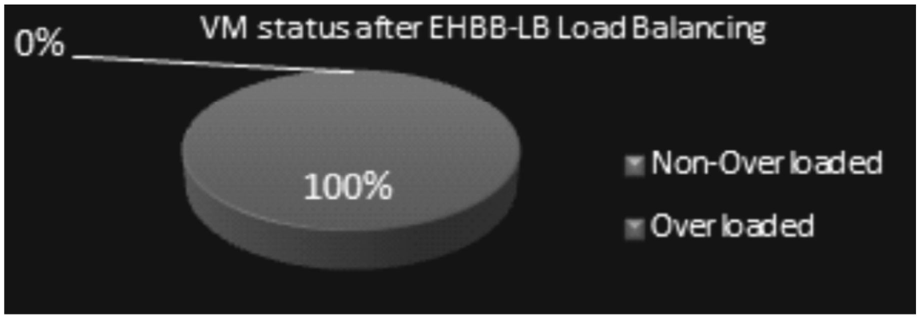

Fig. 6. Percentage of overloaded and non-overloaded VMs after applying E-HBBLB.

In order to compare the ratio of overloaded and non-overloaded VMs after applying the algorithm, we have applied same set of VMs with 40 tasks to HBBLB as well as E-HBBLB. Fig. 2(*a*) shows that after applying HBBLB, there were still 20% of VMs left overloaded whereas after applying E-HBBLB, none of the VMs were overloaded. Fig.2(*b*) depicts that E-HBBLB left the OVM set empty and therefore the percentage of overloaded VMs after applying E-HBBLB is 0%. It reveals that the proposed algorithm clearly outperforms the HBBLB.

7.4 Response Time Vs Number of Task/Instructions

Response time is calculated for FCFS, Honey Bee & Proposed E-HBBLB where numbers of task/ instructions are variable from 5 to 65 tasks.

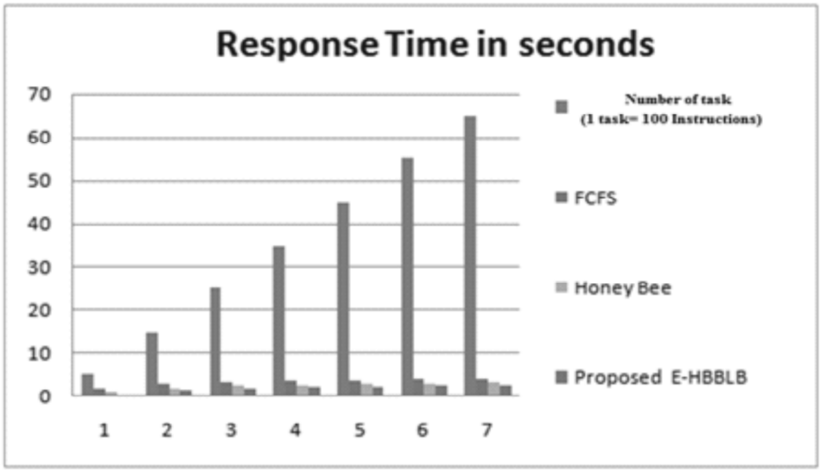

Fig. 7. Response Time Vs Number of Task/Instructions

7.5. Response Time Vs Number of Virtual Machines

Response times is calculated for FCFS, Honey Bee & Proposed E-HBBLB for various numbers of virtual machines from 5 to 30

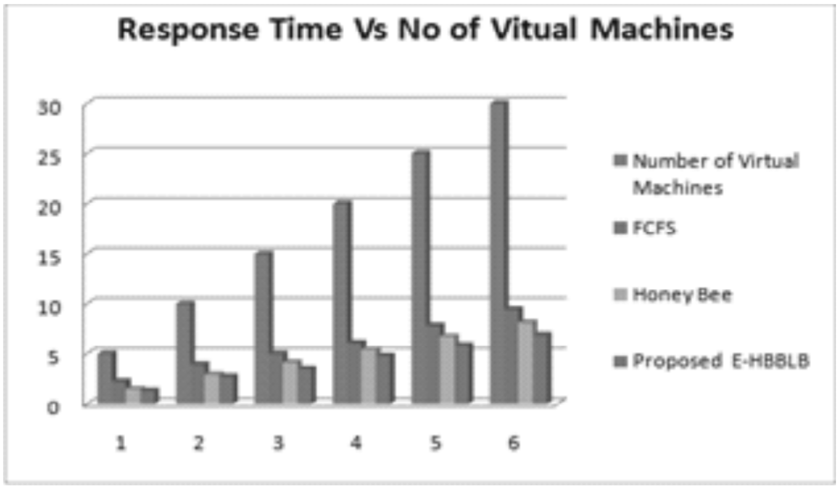

Fig. 8. Response Time Vs Number of Virtual machines.

7.6 Waiting Time

Waiting time is defined as how long each process has to wait before it gets it's time slice. The table 5.5.4 and figure 5.5.4 shown below is the Waiting time to access the resources for the users of the cloud.

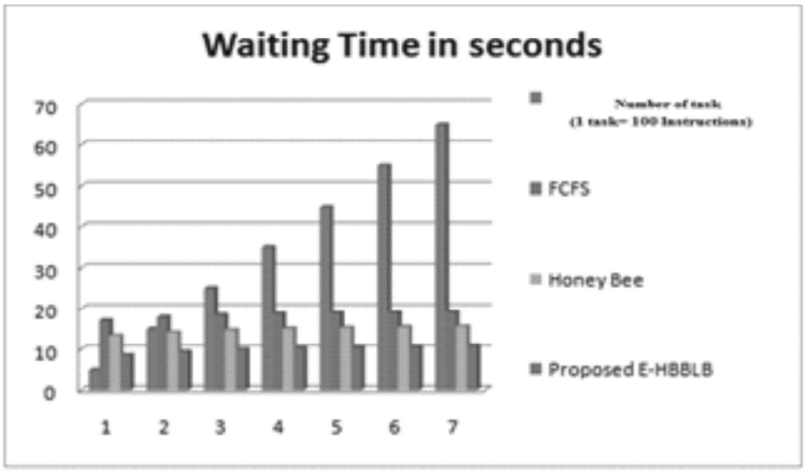

Fig. 9. Waiting Time for FCFS, Honey Bee & Proposed method.

8. CONCLUSION

Cloud Computing offers a very large number of prospects of using IT infrastructure as a utility with a lot of possibilities like scaling down and scaling up depending upon the needs of the association. However, comparable to most rising technologies cloud computing is also having problems that need to be resolved. This work starts with an introduction to cloud computing then explained the difficulties that need to be focused in the coming future as well cover various load balancing techniques, its need and challenges for cloud data.

Present work explains Load Balancing problem as the major problem. The Load balancing method proposed EHBB-LB overcomes the load balancing trouble for cloud computing. The result analysis demonstrates the performance of the proposed methodology and current load balancing methods FCFS and Honeybee method. Various comparison parameters are considered for all the three methods. Simulation results clearly demonstrate that our proposed methodology EHBB-LB having outstanding results in terms of makespan time and execution time as compared with current methods.

9. REFERENCES

1. B. Kruekaew and W. Kimpan, “Virtual Machine Scheduling Management on Cloud Computing Using Artificial Bee Colony”, *Proceedings of the International MultiConference of Engineers and Computer Scientists*, vol. 1 2015
2. F. Ramezani, J. Lu and F. K. Hussain, “Task-Based System Load Balancing in Cloud Computing Using Particle Swarm Optimization”, *Int J Parallel Prog International Journal of Parallel Programming*, vol. 42, no. 5, pp. 739-754, 2013
3. M. Rouse, “What is cloud computing? - Definition from WhatIs.com”, *SearchCloudComputing*, 2015
4. K. Dasgupta, B. Mandal, P. Dutta, J. K. Mandal and S. Dam, “A Genetic Algorithm (GA) based Load Balancing Strategy for Cloud Computing”, *International Conference on Computational Intelligence Modeling Techniques and Applications*, vol. 10, pp. 340-347
5. H. Yuan, C. Li and M. Du, “Optimal Virtual Machine Resources Scheduling Based on Improved Particle Swarm Optimization in Cloud Computing”, *JSW Journal of Software*, vol. 9, no. 3, pp. 705-708, 2014
6. V. Šesum-Èavic, E. Kühn, N. Bessis and F. Xhafa, “Chapter 8 Self-Organized Load Balancing through Swarm Intelligence” in *Next Generation Data Technologies for Collective Computational Intelligence*, vol. 352, pp. 195-224, 2011, Springer
7. S. Mirjalili, “The Ant Lion Optimizer”, *Advances in Engineering Software*, vol. 83, no. 0965¿¿¿9978, pp. 80-98, 2015
8. J. Gu, J. Hu, T. Zhao and G. Sun, “A New Resource Scheduling Strategy Based on Genetic Algorithm in Cloud Computing Environment”, *JCP Journal of Computers*, vol. 7, no. 1, pp. 42-52, 2012
9. S. Singh and M. Kalra, “Task Scheduling Optimization Of Independent Tasks In Cloud Computing Using Enhanced Genetic Algorithm”, *International Journal of Application or Innovation in Engineering & Management*, vol. 3, no. 7, pp. 286-291, 2014
10. M. A. Tawfeek, A. El-Sisi, A. E. Keshk and F. A. Torkey, “Cloud task scheduling based on ant colony optimization”, *2013 8th International Conference on Computer Engineering & Systems (ICCES)*, vol. 12, no. 2, pp. 64-69
11. G. Xu, Y. Ding, J. Zhao, L. Hu and X. Fu, “A Novel Artificial Bee Colony Approach of Live Virtual Machine Migration Policy Using Bayes Theorem”, *The Scientific World Journal*, pp. 1-13, 2013
12. U. Singal and S. jain, “An Analysis of Swarm Intelligence based Load Balancing Algorithms in a Cloud Computing Environment”, *International Journal of Hybrid Information Technology*, vol. 8, no. 1, pp. 249-256, 2015
13. A. Al and F.A. Omara, “Task Scheduling using Hybrid Algorithm in Cloud Computing Environments”, *IOSR Journal of Computer Engineering*, vol. 17, no. 3, pp. 96-106, 2015
14. G. Kaur and S. Sharma, “Research Paper on Optimized Utilization of Resources Using PSO and Improved Particle Swarm Optimization (IPSO) Algorithms in Cloud Computing”, *International Journal of Advanced Research in Computer Science & Technology (IJARCST 2014)*, vol. 2, no. 3, 2014.