Journal of Network and Innovative Computing
ISSN 2160-2174 Volume 4 (2016) pp. 135-142
© MIR Labs, www.mirlabs.net/jnic/index.html

Enhanced Bee Colony Algorithm for Efficient Load Balancing and Scheduling in Cloud

K R Remesh Babu[1], Philip Samuel[2]

[1] Government Engineering College Idukki,
Painavu, Idukki, Kerala - 685603, India
remeshbabu@yahoo.com

[2] Cochin University of Science and Technology,
Kochi - 682022, India
philipcusat@gmail.com

Abstract: **Cloud computing is a promising paradigm which provides resources to customers on their demand with minimum cost. Cost effective optimal scheduling and load balancing are major challenges in adopting cloud computation. Good load balancing and scheduling methods avoids under loaded and heavy loaded conditions in datacenters. When some VMs are overloaded with several number of tasks, these tasks are migrated to the under loaded VMs of the same datacenter in order to maintain Quality of Service (QoS). Frequent VM migrations also affect the performance of the cloud eco system. Nature inspired algorithms are efficient in solving this kind of dynamic problems. This paper proposes a modification in the bee colony algorithm for efficient and effective load balancing in cloud environment. The honey bees foraging behaviour is used to balance load across virtual machines. The tasks removed from over loaded VMs are treated as honeybees and under loaded VMs are the food sources. The method also tries to minimize makespan as well as number of VM migrations. The algorithm also tries to reduce the imbalance in the cloud eco system. The experimental result shows that there is significant improvement in the QoS delivered to the customers.**

Keywords: Cloud computing, Task Scheduling, Bee colony algorithm, Load balancing, Imbalance, QoS.

I. Introduction

Cloud computing is an emerging technology completely relying on internet, in which all the data and applications are hosted on datacenters, which consists of thousands of computing resources interlinked together in a complex manner. The cloud providers adopt pay as you use model for their resource utilization. Over the Internet, the customers can use computation power, software resources, storage space, etc., by paying money only for the duration he has used the resource.

Besides Internet, customers, datacenters, and distributed servers are the main three components of a cloud eco system. Datacenter is a collection of servers hosting different applications and also provides storage facility. In order to subscribe for different applications, end user needs to connect to the datacenter. Usually a datacenter is situated far away from the end users. Distributed servers are the parts of a cloud environment which are present throughout the Internet hosting different applications.

In order to ensure QoS efficient scheduling and load balancing among nodes are required in the distributed cloud environment. In cloud computing ensuring QoS is crucial for customer satisfaction. An efficient load balancing mechanism tries to speed up the execution time of user requested applications. It also reduces system imbalance and gives a fair response time to the users.

Better load balancing will result in good QoS metrics such as efficient resource utilization, scalability, response time, and fault tolerance. Also migration time can be improved by better load balancing. The improvement in the above factors will ensure good QoS to the customers thereby less Service Level Agreement (SLA) violations.

Static load balancing algorithms will work only when there is a small variation in the workload. Cloud scheduling and load balancing problems are considered as NP hard problems. The dynamic nature of cloud computing environment need dynamic algorithms for efficient and effective scheduling and load balancing among computing nodes.

A. *Nature Inspired Algorithms*

Nature inspired algorithms and swarm intelligence algorithms play a vital role in solving dynamic real time problems, which are hard to solve by normal methods. These NP hard problems are hard to solve within a time limit. Nature inspired and swam intelligence based algorithms produce optimal or near optimal solutions to these real time problems in polynomial time interval. The idea behind swarm intelligence algorithm is that local interaction of many simple agents to attain a simple objective. Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), Cuckoo Search and Artificial Bee Colony (ABC) algorithm are some of the algorithms in this category.

These algorithms can be combined with one another for solving dynamic problems in cloud environment. In [24] Fish Swarm Search (FSS) algorithm and PSO are combined to solve scheduling problem in cloud environment. ACO and ABC algorithm are merged in [17] to solve dynamic grid task scheduling. Although there are several nature inspired algorithms, but all are not suitable for all situations.

B. *Bee Colony Algorithm*

The Bee Colony algorithm is a meta heuristic swarm intelligence algorithm [1] based on the foraging behavior of honey bee colonies to solve numerical function optimization problems. It mimics the foraging behavior of honey bees. It has advantages such as memory, multi-character, local search and solution improvement mechanism, so it is an excellent solution for optimization problems [16][17][18].

The Bee colony model is composed of three main elements: the employed and unemployed foragers are the first two elements, while the third element is the rich food sources close to their hive. The two leading modes of behavior are also described by the model. These behaviors are necessary for self-organization and collective intelligence: recruitment of forager bees to rich food sources, resulting into positive feedback and simultaneously, the abandonment of poor sources by foragers, which causes negative feedback.

The Bee Colony consists of three groups of artificial bees: employed foragers, onlookers and scouts. The employed bees comprise the first half of the colony whereas the second half consists of the onlookers. The employed bees are linked to particular food sources. In other words, the number of employed bees is equal to the number of food sources for the hive. The onlookers observe the dance of the employed bees within the hive, to select a food source, whereas scouts search randomly for new food sources.

The search cycle of Bee Colony consists of three rules:

- Sending the employed bees to a food source and evaluating the nectar quality
- Onlookers choosing the food sources after obtaining information from employed bees and calculating the nectar quality
- Determining the scout bees and sending them onto possible food sources

The positions of the food sources are randomly selected by the bees at the initialization stage and their nectar qualities are measured. The employed bees then share the nectar information of the sources with the bees waiting at the dance area within the hive. After sharing this information, every employed bee returns to the food source visited during the previous cycle, since the position of the food source had been memorized and then selects another food source using its visual information in the neighbourhood of the present one.

At the last stage, an onlooker uses the information obtained from the employed bees at the dance area to select a food source. The probability for the food sources to be selected increases with increase in its nectar quality. Therefore, the employed bee with information of a food source with the highest nectar quality recruits the onlookers to that source. It subsequently chooses another food source in the neighbourhood of the one currently in her memory based on visual information (i.e. Comparison of food source positions). A new food source is randomly generated by a scout bee to replace the one abandoned by the onlooker bees. This search process is represented below [19]:

```
1. Initialize the Bee Colony and problem parameters
2. Initialize the Food Source Memory (FSM)
3. Send the employed bees to the food sources.
4. Send the onlookers to select a food source.
5. Send the scouts to search possible new food.
6. Memorize the best food source.
7.Repeat the steps 3-6 until termination criterion are met
```

Figure 1. Bee Colony Algorithm

The proposed algorithm consists of scout bees, forager bees and food source. In bee hives, scout bees forage for food sources. After finding a food source it returned to bee hive and performs a waggle dance. Based on waggle dance other bees in the hive get information about quantity of food and distance from the bee hive. Then forager bees follow the scout bees to the location of bee hive and begin to reap it. The positions of food sources are randomly selected by the bees.

In the proposed method, bee colony algorithm is modified and it is applied to efficiently schedule and balance the load among computing nodes in the dynamic cloud environment. This method considers previous state of a node while distributing the workload. For load balancing the bee colony algorithms parameters are mapped to cloud environment. The algorithm tries to achieve minimum response time and completion time. The remaining part of this paper is organized as follows. Section 2 describes about different kinds of load balancing methods in cloud. Enhanced bee colony algorithm and its architecture described in Section 3. The section 4 gives experimental results and analysis. Finally this paper concludes in section 5.

II. Related Works

Efficient scheduling and load balancing ensures better QoS to the customers and thereby reduces number of SLA violations. This section reviews some of the load balancing algorithms.

Modified throttled algorithm based load balancing is presented in [2]. While considering both availability of VMs for a given request and uniform load sharing among the VMs for number of requests served, it is an efficient approach to handle load at servers. It has an improved response time, compared to existing Round-Robin and throttled algorithms.

In [3], a load balancing approach was discussed, which manages load at server by considering the current status of all available VMs for assigning the incoming requests. This VM-assign load balancing technique mainly considers efficient utilization of the resources and VMs. By simulation, they proved that their algorithm distributes the load optimally and hence avoids under / over utilization of VMs. The comparison of this algorithm with active-VM load balance algorithm shows that their algorithm solves the problem of inefficient utilization of the VMs.

Response time based load balancing is presented in [4]. In order to decide the allocation of new incoming requests, proposed model considers current responses and its variations. The algorithm eliminates need of unnecessary communication of the Load Balancer. This model only considers response time which is easily available with the Load Balancer as each request and response passes through the Load Balancer, hence eliminates the need of collecting additional data from any other source thereby over utilizing the communication bandwidth.

In [5] a load balancing technique for cloud datacenter, Central Load Balancer (CLB) was proposed, which tried to avoid the situation of over loading and under loading of virtual machines. Based on priority and states, the Central Load Balancer manages load distribution among various VMs. CLB efficiently shares the load of user requests among various virtual machines.

Ant colony based load balancing in cloud computing was proposed in [6]. It works based on the deposition of pheromone. A node with minimum load is attracted by most of the ants. So maximum deposition of pheromone occurs at that node and performance is improved.

Cloud Light Weight (CLW) for balancing the cloud computing environment workload is presented in [7]. It uses two algorithms namely, receiver-initiated and sender-initiated approaches. VM Attribute Set is used to assure the QoS. CLW uses application migration (as the main solution) instead of using VM migration techniques in order to assure minimum migration time.

A resource weight based algorithm called Resource Intensity Aware Load balancing (RIAL) is proposed in paper [8]. In this method, VMs are migrated from over-loaded Physical Machines (PM) to lightly loaded PMs. Based on resource intensity the resource weight is determined. A higher-intensive resource is assigned a higher weight and vice versa in each PM. The algorithm achieves lower-cost and faster convergence to the load balanced state, and minimizes the probability of future load imbalance, by considering the weights when selecting VMs to migrate out and selecting destination PMs.

A cloud partitioning based load balancing model for public cloud was proposed in [9]. This algorithm applies game theory to load balancing strategy in order to improve the efficiency. Here a switching mechanism is used to choose different strategies for different situations.

Time and cost based performance analysis of different algorithms in cloud computing was given in [10]. A load balancing mechanism based on artificial bee colony algorithm was proposed in [11]. It optimizes the cloud throughput by mimicking the behavior of honey bees. Since bee colony algorithm arranges only a little link between requests in the same server queue, then maximization of the system throughput is suboptimal. Here, the increasing request does not leads to the increase of system throughput in certain servers.

An active clustering based load balancing technique is presented in paper [12]. It groups similar nodes together and works on these groups and produces better performance with high utilization of resources. Paper [13] proposes a Best-fit – Worst-fit strategy that efficiently places the virtual machines to the lesser number of active PMs. In this two level scheduling mechanism the tasks are scheduled using best-fit approach. Then the cloud broker uses worst-fit method for VM placement. They have considered cost and energy for the effective placement of VMs.

Weighted Signature based Load Balancing (WSLB), a new VM level load balancing algorithm is presented in [14]. This algorithm find the load assignment factor for each host in a datacenter and map the VMs according to that factor. Estimated finish time [15] based load balancing considers the current load of virtual machines in a datacenter and the estimation of processing finish time of a task before any allocation. This algorithm improves performance, availability and maximizes the use of virtual machines in their datacenters. In order to avoid a probable blocking of tasks in the queue, it permanently controls current load on the virtual machines and the characteristics of tasks during processing and allocation.

The authors in [25] proposed a heuristic based scheme for load balancing in the large cloud data centers based on duplicating jobs and sending replicas to different servers. They showed that this mechanism can significantly reduce the queuing time, even with a small number of replicas and in particular in high workloads. Determining the right parameter configuration for this method (the number of replicas, the server job selection policy) is highly dependent on the system condition, comprising the scale of the system, load pattern, job processing time, and inter-server delays. As different systems may be subject to different conditions, there is no single parameter configuration that is optimal to all systems. So inorder to deploy the proposed scheme, the system manager should conduct a simulation-based study to determine the right settings for the specific system. But cloud is a dynamic environment, so the conditions may change. Therefore system performance should be constantly monitored in order to determine whether any of the parameter values should be modified.

A dynamic load-balanced scheduling (DLBS) based on heuristic algorithms approach to maximize the network throughput through dynamically balancing data flows is developed in [26]. In this method the data flow is balanced time slot by time slot. The simulation result shows that the algorithm works better when data flow is high.

There are several methods proposed for load balancing in cloud such as collaborative agents for distributed problem solving [27], CLB load balancing architecture and algorithm [28], Temporal task scheduling with heuristics [29], QoS based methods [30], and concave pricing [31]. In this most of them are single objective in nature.

III. Bee Colony based Load Balancing Algorithm

As more and more users enters into cloud computing, load balancing is a significant task in resource management. An optimal task scheduling algorithm is able to handle the load balancing problems as well as users' expectations in QoS. The paper [21] proposed a good load balancing algorithm in cloud environment. The Interaction ABC (IABC) algorithm proposed in this paper enhances the production of the systems and schedule the tasks to virtual machines (VMs) more efficiently. The authors simulated algorithm and results shows that finishing time of all tasks in the same system will be less than others' using this method. The papers [22] and [23] also tried bee colony algorithm for load balancing in cloud environment.

The paper [20] developed a load balancing and scheduling algorithm based on bee colony for cloud environment. This algorithm is suitable for dynamic environment, because it converges to the optimal solution within in little iterations. The paper considered completion time of tasks, and number of task migrations during computation. The algorithm tries to reduce number of migrations.

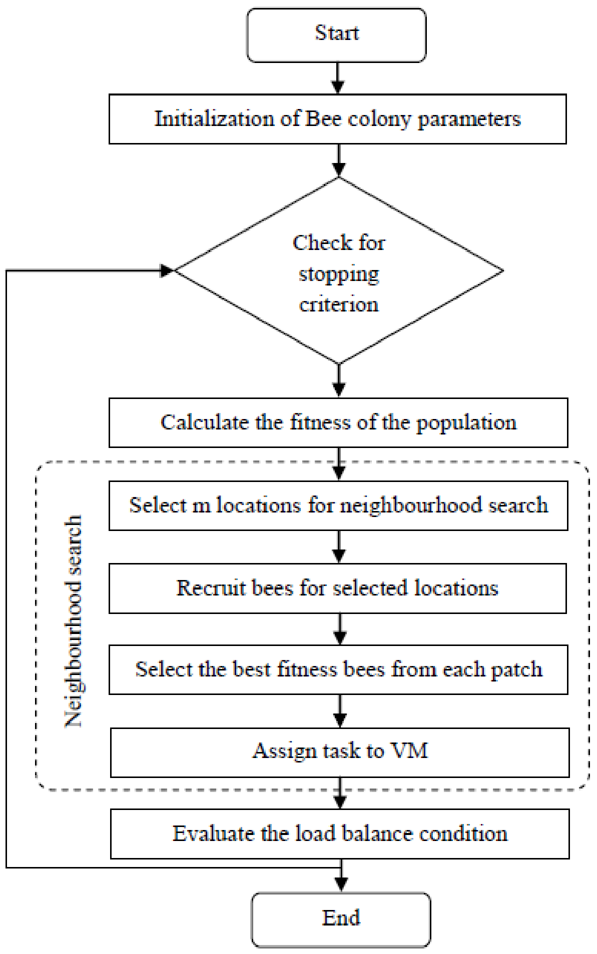
Start
Initialization of Bee colony parameters
Check for stopping criterion
Calculate the fitness of the population
Neighbourhood search
Select m locations for neighbourhood search
Recruit bees for selected locations
Select the best fitness bees from each patch
Assign task to VM
Evaluate the load balance condition
End

Figure 2. Load balancing using bee colony algorithm

The basic steps used in bee colony algorithm is given in figure 3.

1. Start
2. Find load of each VMs and group VMs as over-loaded or under loaded.
3. Find the supply of under loaded VMs and demand of overloaded VMs.
4. Sort the overloaded and under loaded VM sets
5. Sort the tasks in overloaded VMs based on priority.
6. For each task in each overloaded VM find a suitable under loaded VM.
7. Update the overloaded and under loaded VM sets and go to step 2.
8. Stop

Figure 3. Basic Bee Colony Algorithm for Cloud load balancing

IV. Modified Bee Colony Algorithm for Load Balancing

The proposed method uses the foraging behaviour of honeybees for effective load balancing across VMs and reschedules the cloudlets into under loaded VMs. For efficient implementation of honey bee algorithm, the foraging behaviour of honeybees is mapped into cloud environment in order to achieve load balancing. The mapping of bee colony parameters with cloud environment is given in Table 1.

Table 1. Mapping of Enhanced Bee colony parameters with Cloud environment

Honey Bee Hive	Cloud Environment
Honey bee	Task (Cloudlet)
Food source	VM
Honey bee foraging a food source	Loading of a task to a VM
Honey bee getting depleted at a food source	VM in overloaded condition
Foraging bee finding a new food source	Removed task will be rescheduling to an under loaded VM having highest capacity

In this proposed method the tasks are considered as honeybees. When the honeybees forage for food source, the cloudlets will be assigned in VMs for execution. Since the processing capacity varies for different VMs, sometimes VMs may be overloaded and others will be under loaded. In these circumstances in order to provide better performance and efficient load balancing mechanism is needed. When a particular VM is overloaded with multiple tasks then some tasks are need to be migrated and have to assign to an under loaded VM. In this case, task to be migrated is chosen based on priority. In the proposed method tasks with lowest priority will be selected as a candidate for migration. This procedure is similar as honey is exhausted in nectar and bees are ready to take off from the food source.

The architecture for the proposed load balancing method is given in figure. 4. Cloud Information Service (CIS) is the repository that contains all the resources available in the cloud environment. It is a registry of datacenters. When a datacenter is created it has to register to the CIS. Datacenters are heterogeneous in nature with specific characteristics. Usually a datacenter consists of several hosts. Hosts have number of processing elements (PEs) with RAM and bandwidth characteristics. In cloud environment these hosts are virtualized into different number of VMs based on user request. VMs may also have heterogeneous characteristics like hosts.

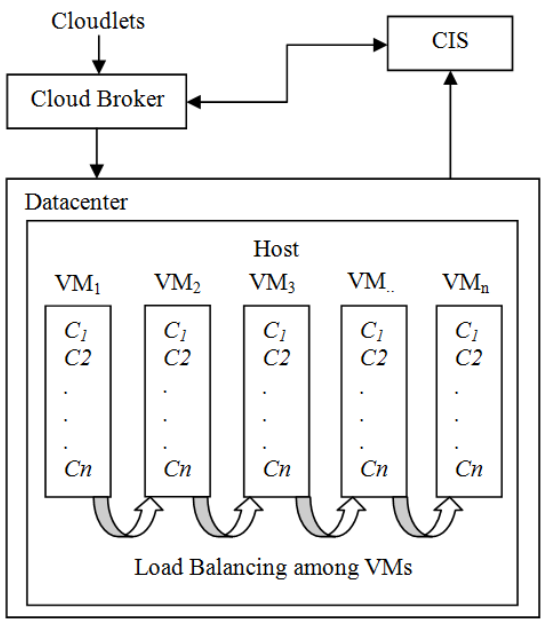

Figure 4. Load balancing architecture

CIS collect information about the all resources in the datacenters. Based on this information the cloud broker submits these tasks to different VMs in a datacenter. In the proposed method the algorithm checks for overloaded conditions and it migrates task from overloaded VMs to under loaded VMs. The enhanced bee colony algorithm is given in figure 5.
The proposed load balancing and scheduling mechanism works in four different steps as given below.

1. VM Current Load Calculation
2. Load Balancing & Scheduling Decision
3. VM Grouping
4. Task Scheduling

1) VM Current Load Calculation

The current load on a VM is measured based on the ratio between total lengths of the tasks submitted to that VM to the processing rate of that VM at a particular instance. Suppose N is the total numbers tasks assigned to a VM and *Len* is the length of single tasks and MIPS is the Million Instruction Per Second rate of that VM, then using the equation (1) the current load can be calculated.

$$Load_{VM} = \frac{N*Len}{MIPS} \quad (1)$$

Then total load on a datacenter is the sum of load on each VMs. The equation for total load a datacenter $Load_{DC}$ is given by the equation (2).

$$Load_{DC} = \sum_{vm=1}^{n} Load_{VM} \quad (2)$$

The processing capacity of VM can be calculated using the equation (3) as given below.

$$Capacity_{VM} = PE_{num} * PE_{mips} + VM_{bw} \quad (3)$$

Here PE_{num} is the number of processing elements in a particular VM, PE_{mips} is the processing power of PE in MIPS rate and VM_{bw} is the band width associated for a VM.
A datacenter may have several VMS. So the total capacity of the entire datacenter can be calculated from using the equation (4),

$$Capacity_{DC} = \sum_{vm=1}^{n} Capacity_{VM} \quad (4)$$

Then the proposed algorithm computes processing time of each task using equation (5).

$$PT = \frac{Current\ Load}{Capacity} \quad (5)$$

Then the processing time required for datacenter to complete all the tasks in it can be calculated by the equation (6) given below,

$$PT_{DC} = \frac{Load_{\ DC}}{Capacity_{\ DC}} \quad (6)$$

Then the Standard Deviation (SD) is a good measure of deviations. The proposed method uses SD for measuring the deviations in the workload on each VM. Equation (7) gives the SD of loads.

$$SD = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(PT_i - PT)^2} \quad (7)$$

Then the load balancing decision is done based on the value of SD.

```
1. Start
2. For each task do
3. Calculate the load on VM and decide whether to do
   load balancing or not
4. Group the VMs based on load as overloaded or under
   loaded.
5. Find the supply of under loaded VMs and demand of
   overloaded VMs.
6. Sort the overloaded and under loaded VM sets
7. Sort the tasks in overloaded VMs based on priority.
8. Find the capacity of VMs in the under loaded set.
9. For each task in each overloaded VM find a suitable
   under loaded VM based on capacity.
10. Update the overloaded and under loaded VM sets
11. End of step 2.
12. Stop
```

Figure 5. Enhanced Bee colony based load balancing algorithm

In this proposed method bee colony algorithm is modified to find optimal solution quickly. This algorithm quickly converges into an optimal solution. The algorithm also tries to minimize the number of task migrations. It also considers users priority while scheduling the tasks.

2) Load Balancing & Scheduling Decision

In this phase, load balancing and rescheduling of tasks are decided. This decision depends on the SD value calculated using equation (7). In order to maintain system stability the load balancing and scheduling decision will take only when the capacity of the datacenter is greater than current load. Otherwise it will create imbalance in the datacenter. For finding the load a threshold value is set (value lies in 0-1) based on the SD calculated. The systems compare this value with calculated SD measure. The load balancing and scheduling done only if the calculated SD is greater than the threshold. This will improve the system stability by minimizing number of migrations.

3) VM Grouping

In order to increase the efficiency VMs are grouped into two groups: overloaded VMs and under loaded VMs. This will reduce the time required to find optimal VM for task migration. The overloaded VMs are the candidates for migration. In the proposed method these removed tasks are considers as honeybees and the under loaded VMs are their food sources. The VMs are grouped according to the SD and threshold value already calculated based on the load.

4) Task Scheduling

Before initiating load balancing the system have to find the demand to each overloaded VMs and supply to the under loaded VMs. Here the VMs are sorted based on the capacity in ascending order. The task migration is performed only when demand meets the supply. From the under loaded VM set, the proposed method selects a VM which has highest capacity as target VM. The method selects the task with lowest priority from an overloaded VM and it is rescheduled to an under loaded VM with maximum capacity.

Supply to a particular VM is the difference between its capacity and current load and it can be calculated using equation (8),

$$Supply_{VM} = Capacity - Load \quad (8)$$

Then the demand of a VM is calculated using the equation (9)

$$Demand_{VM} = Load - Capacity \quad (9)$$

On submission of each task into the cloud, the VM will measure the current load status and calculates SD. If the SD of loads is greater than the threshold then load balancing process is initiated. During this load balancing process, VMs are classified into under loaded and overloaded VM sets. Then the submitted tasks are rescheduled to the VM having highest capacity.

V. Experimental Results

The performance analysis of the proposed method is carried out in a simulated dynamic environment. In this heterogeneous environment VMs having different specification are deployed. Cloudlets with varying specifications are submitted into this cloud environment. The number of VM migrations and makespan are measured and compared with existing methods.

The makespan time of the proposed enhanced bee colony method with bee colony algorithm is shown in Table 2. The overall task completion time, i.e. makespan is graphically represented in Fig. 6. From these results it is clear that makespan is reduced into a significant amount while using enhanced bee colony algorithm. Then the users will get faster response than older methods. Response time is a good measure of QoS provided by the service provider. So here the provider can assure good QoS to their customers.

Table 2. Comparison of Makespan

Number of Cloudlets	Bee Colony (Sec)	Enhanced Bee Colony (Sec)
10	50.10	43.85
15	70.10	68.85
20	80.10	78.85
25	110.10	100.10
30	120.10	118.85

If number of task migrations is greater it will adversely affects the performance of the cloud and thereby reduces the QoS. Frequent migrations will adversely affect the system stability of the cloud environment. A good load balancing and scheduling mechanism will reduce the number of task migrations. In this method the algorithm consider the priorities of the tasks when a migration is needed. Lower priority task will have higher chance for migration to the under loaded servers, so that higher priority customers are unaffected. The proposed method is analyzed for number of task migrations. The results are tabulated in the Table 3.

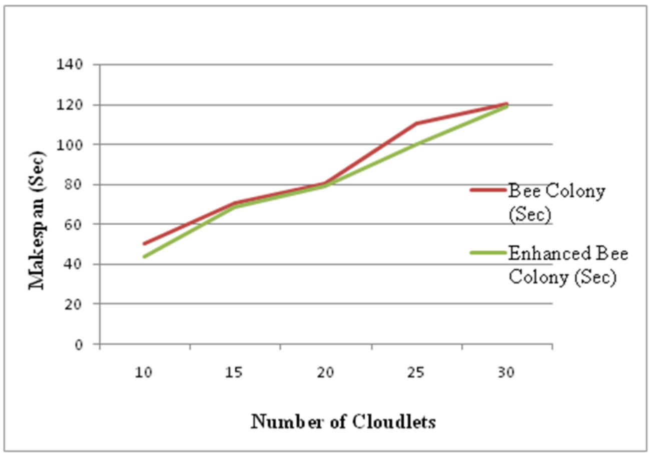

Figure 6. Comparison of Makespan

The result in Table 3 shows that the number of task migrations is reduced while using enhanced bee colony algorithm. In most of the cases the algorithm out performs the existing bee colony algorithm. If frequent migration of tasks are happened it will adversely affect the performance of the entire cloud eco system and thereby its performance.

Table 3. Comparison of Task Migration

Number of Cloudlets	Bee Colony	Enhanced Bee Colony
10	2	2
15	4	3
20	7	7
25	12	11

The above experimental results shows that how the proposed enhanced honey bee algorithm reduces the makespan as well as number of task migrations compared to the existing bee colony algorithm. Thus it helps efficient and effective use of computational resource in the cloud environment. Since the algorithm minimizes the completion and reduces number of task migrations it will give better QoS to the end users. The number of task migration is given in the Fig. 7.

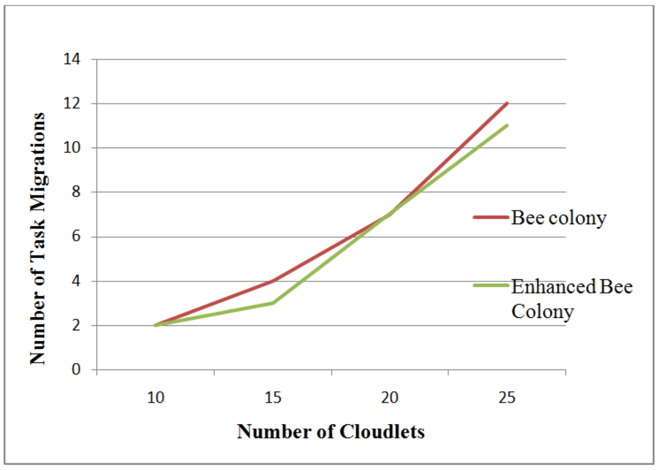

Figure 7. Number of task migrations

In order to measure the impact of VM migrations here we introduced a term the degree of imbalance for cloud computing. . It can be defined as the ratio of difference between maximum and minimum execution time to the average execution time. Because frequent VM migrations affects the completion time of a task.

Table 4. Degree of Imbalance

Number of Cloudlets	Before	After
10	1.000	0.714
15	0.509	0.664
20	0.840	0.268
25	0.667	0.212

Table 4 and Figure 8 represent the degree of imbalance before and after load balancing. From the figure it is clear that the imbalance is reduced to a considerable amount after load balancing.

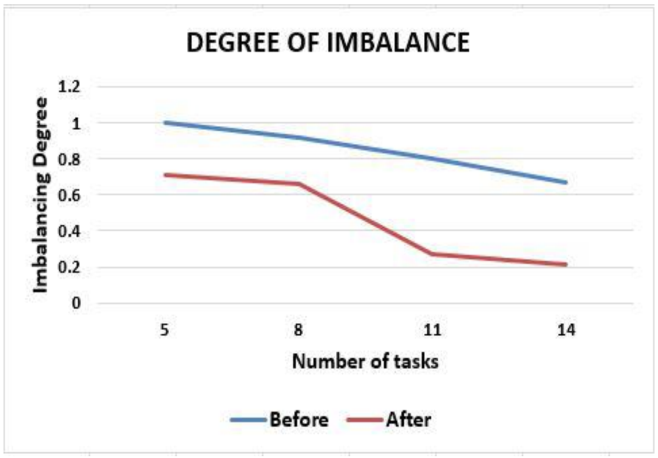

Figure 8. Degree of Imbalance Before and After

VI. Conclusion

Nature inspired algorithms are good solution provider for real time dynamic optimization problems. This paper proposes an enhanced bee colony algorithm for efficient load balancing in cloud environment. Here the power of swarm intelligence algorithm is used to remove the tasks from overloaded VMs and submitted this removed tasks to the most appropriate under loaded VMs. It not only balances the load, but also considers the priorities of tasks in the waiting queues of VMs. The task with least priority is selected for migration to reduce imbalance. So no tasks are needed to wait longer time in order to get processed. The experimental results show that, the proposed algorithm outperforms existing bee colony algorithm and minimize makespan and number of migrations and gives better QoS to end users.

In future the algorithm can be further enhanced with hybridization of other nature inspired algorithms like Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), etc.

References

[1] Dervis Karaboga, Bahriye Basturk. “A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm”, *Springer, J Glob Optim*, vol. 39, pp. 459–471, 2007

[2] Domanal., Shridhar, G.R., Ram Mohana, G. “Load Balancing in Cloud Computing Using Modified Throttled Algorithm”, *In Proceedings of IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)*, pp. 1-5, 2013

[3] Shridhar, G.D., G.R.M., Reddy. “Optimal Load Balancing in Cloud Computing By Efficient Utilization of Virtual Machines”, *In Proceedings of IEEE Sixth International Conference on Communication Systems and Networks (COMSNETS)*, pp. 1-4, 2014

[4] Agraj Sharma, Sateesh K., Peddoju. “Response Time Based Load Balancing in Cloud Computing”, *In Proceedings of International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)*, pp. 1287-1293, 2014

[5] Soni. G., Kalra. M. “A Novel Approach for Load Balancing in Cloud Data Center”, *In Proceedings of IEEE International Conference on Advance Computing Conference (IACC)*, pp. 807-812, 2014

[6] Santanu Dam, Gopa Mandal, Kousik Dasgupta, Paramartha Dutta. “An Ant Colony Based Load Balancing Strategy in Cloud Computing”, *Springer Advanced Computing, Networking and Informatics*, Vol. 28, pp. 403-413, 2014

[7] Mohammadreza M., Amir M.R., Anthony T.C. “Cloud Light Weight: a New Solution for Load Balancing in Cloud Computing”, *In Proceedings of International Conference on Data Science & Engineering (ICDSE)*, pp. 44-50, 2014

[8] L Chen, H Shen, K Sapra. “RIAL: Resource Intensity Aware Load Balancing in Clouds”, *In Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, pp. 1294-1302, 2014

[9] G Xu, J Pang, Xiaodong Fu. “A Load Balancing Model Based on Cloud Partitioning for the Public Cloud”, *Journal of Tsinghua Science and Technology*, pp. 34-39, 2013

[10] M. Randles, D. Lamb, A. Taleb-Bendiab. “A comparative study into distributed load balancing algorithms for cloud computing”, *In Proceedings of IEEE 24th International Conference on Advanced Information Networking and Applications*, Perth, Australia, pp. 551-556, 2010

[11] Jing Yao, Ju-hou He. “Load Balancing Strategy of Cloud Computing based on Artificial Bee Algorithm”, *In Proceedings of IEEE 8th International Conference on Computing Technology and Information Management (ICCM)*, pp. 185-189, 2012

[12] Pooja Samal. “Analysis of variants in Round Robin Algorithms for load balancing in Cloud Computing”, *International Journal of Computer Science and Information Technologies*, vol. 4 (3), pp. 416-419, 2013

[13] Remesh Babu, K.R., Philip Samuel. “Virtual Machine Placement for Improved Quality in IaaS Cloud”, *In Proceedings of IEEE Fourth International Conference on Advances in Computing and Communications (ICACC)*, pp. 190-194, 2014

[14] Ajit. M., Vidya. G., “VM Level Load Balancing in Cloud Environment”, *In Proceedings of IEEE Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, pp. 1-5, 2013

[15] Fahim, Y., Ben Lahmar E., El Labrlji E.H., Eddaoui A. “The load balancing based on the estimated finish time of tasks in cloud computing”, *In Proceedings of 2nd World Conference on Complex Systems (WCCS)*, pp. 594-598, 2014
[16] Madivi R., S Sowmya Kamath. “An hybrid bio-inspired task scheduling algorithm in cloud environment”, *In Proceedings of International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pp. 1 -7, 2014
[17] Remesh Babu, K.R., P Mathiyalagan, S.N., Sivanandam. “Pareto based hybrid Meta heuristic ABC – ACO approach for task scheduling in computational grids”, *International Journal of Hybrid Intelligent Systems*, Vol. 11, No. 4/2014, pp. 241-255, 2014
[18] Ling Wang, Gang Zhou, Y Xu, M Liu. “An enhanced Pareto-based artificial bee colony algorithm for the multi-objective flexible job-shop scheduling”, *Int. J. Adv. Manufacturing Technology*, vol. 60, Issue 9-12, pp. 1111-1123, 2012
[19] http://mf.erciyes.edu.tr/abc/.: Artificial Bee Colony Algorithm. Home page, Retrieved on 15th January 2016.
[20] Remesh Babu K.R., Philip Samuel. “Enhanced Bee Colony Algorithm for Efficient Load Balancing and Scheduling in Cloud”, *Advances in Intelligent Systems and Computing*, Vol. 424, pp. 67-78, 2015
[21] Jeng-Shyang Pan , Haibin Wang, Hongnan Zhao, Linlin Tang. “Interaction Artificial Bee Colony Based Load Balance Method in Cloud Computing”, *Genetic and Evolutionary Computing, Vol. 329 of the series Advances in Intelligent Systems and Computing*, pp. 49-57, 2015
[22] Dhinesh Babu L.D., P. Venkata Krishna. “Bee behavior inspired load balancing of tasks in cloud computing environments”, *Applied Soft Computing*, Vol. 13, Issue 5, pp. 2292–2303, 2013
[23] Jing Yao, Ju-hou He. “Load balancing strategy of cloud computing based on artificial bee algorithm”, *Proceedings of 8th International Conference on Computing Technology and Information Management (ICCM)*, Vol.1, pp.185-189, 2012
[24] K R Remesh Babu, Alia Teresa T M, A Neela Madheswari, Philip Samuel. “Improved FSS algorithm with QoS in IaaS Cloud”, *Int. J. Advanced Research in Engineering and Technology*, vol.5, issue 2, pp.138-151, 2014
[25] Amir Nahir, Ariel Orda, and Danny Raz. "Replication-Based Load Balancing", *IEEE Transactions on Parallel and Distributed Systems*, Vol. 27, No.2, pp. 494-507, 2016
[26] Feilong Tang, Laurence T. Yang, Can Tang, Jie Li, and Minyi Guo. "A Dynamical and Load-Balanced Flow Scheduling Approach for Big Data Centers in Clouds", *IEEE Transactions on Cloud Computing*, DOI 10.1109/TCC.2016.2543722, pp. 1-14, 2016
[27] J. Octavio Gutierrez-Garcia and Adrian Ramirez-Nafarrate. "Collaborative Agents for Distributed Load Management in Cloud Data Centers Using Live Migration of Virtual Machines", *Transactions on Services Computing*, vol.6, no.8, pp. 916-929, 2015
[28] Shang-Liang Chen, Yun-Yao Chen, Suang-Hong Kuo. "CLB: A novel load balancing architecture and algorithm for cloud services", *Computers & Electrical Engineering*, doi:10.1016/j.compeleceng.2016.01.029, 2016
[29] Haitao Yuan, Jing Bi, Wei Tan, and Bo Hu Li. "Temporal Task Scheduling With Constrained Service Delay for Profit Maximization in Hybrid Clouds", *IEEE Transactions on Automation Science and Engineering*, DOI: 10.1109/TASE.2016.2526781, Vol. PP, No. 99, pp. 1-12, 2016
[30] W. K. Hsieh, W. H. Hsieh, J. L. Chen, P. J. Yang. "CssQoS: A load balancing mechanism for cloud serving system", *Information Technology and Applications*, DOI: 10.1201/b18284-55, pp.269-275, CRC Press 2015
[31] Rui Zhang, Kui Wu, Minming Li, and JianpingWang. "Online Resource Scheduling Under Concave Pricing for Cloud Computing", *IEEE Transactions on Parallel and Distributed Systems*, Vol. 27, No.4, pp. 1131-1145, 2016

Author Biographies

K R Remesh Babu received his BSc. degree in Mathematics from Mahatma Gandhi University, Kottayam, India and B.Tech in IT from CUSAT, Kochi, India and ME in Computer Science from Anna University, Chennai, India. He is currently pursuing the Ph.D. degree at CUSAT. He is an assistant professor in department of Information Technology, Government Engineering College Idukki, India. His research interests includes Distributed Computing, Cloud computing, Internet of Things, Wireless Sensor Networks, and Big Data Analytics.

Philip Samuel is Reader in Information Technology Division, School of Engineering, Cochin University of Science & Technology (CUSAT). He holds a Masters Degree in Computer & Information Science from CUSAT and Ph.D degree in Computer Science & Engineering from IIT Kharagpur, India. He has more than 17 years of experience in teaching and research as a faculty at CUSAT. He has published more than 35 research papers in International Conferences and Journals. His research interest includes Big Data Analytics, Distributed Computing and Automated Software Engineering