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

K.R. Remesh Babu and Philip Samuel

Abstract Cloud computing is a promising paradigm which provides resources to customers on their request with minimum cost. Cost effective scheduling and load balancing are major challenges in adopting cloud computation. Efficient load balancing 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). 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 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 · Qos

1 Introduction

Cloud computing is an emerging technology completely rely on internet, in which all the data and applications are hosted on a datacenters, which consists of thousands of computers interlinked together in a complex manner. The cloud providers adopt pay as you use model for their resource utilization. Over the Internet, the

K.R. Remesh Babu (✉)
Government Engineering College, Idukki Painavu, Kerala, India
e-mail: remeshbabu@yahoo.com

P. Samuel
Cochin University of Science and Technology, Kochi, India
e-mail: philipcusat@gmail.com

V. Snášel et al. (eds.), *Innovations in Bio-Inspired Computing and Applications*,
Advances in Intelligent Systems and Computing 424,
DOI 10.1007/978-3-319-28031-8_6

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 access to the users.

Better load balancing will result in good QoS metrics such as efficient resource utilization, scalability, response time, 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.

The dynamic nature of cloud computing environment needs a dynamic algorithms for efficient and efficient scheduling and load balancing among nodes. Static load balancing algorithms will works only when small variation in the workloads. Cloud scheduling and load balancing problems are considered as NP hard problems.

Nature inspired algorithms plays 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 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.

The Bee Colony algorithm is a 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 [2–4]. The 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 cloud nodes in the dynamic cloud

environment. Here this method considers previous state of a node while distributing the load. For load balancing the bee colony algorithms parameters are mapped to cloud environment for achieving load balancing. 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 Sect. 3. The Sect. 4 gives experimental results and analysis. Finally this paper concludes in Sect. 5.

2 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 [5]. 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 [6], 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 [7]. 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 wasting the communication bandwidth.

In [8] 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 [9]. 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 [10]. 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 [11]. In this method, VMs are migrated from overloaded 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 [12]. This algorithm applies game theory to load balancing strategy in order to improve efficiency. Here a switch 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 [13]. A load balancing mechanism based on artificial bee colony algorithm was proposed in [14]. 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 [15]. It groups similar nodes together and works on these groups and produces better performance with high utilization of resources. Paper [16] 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 [17]. This algorithm find the load assignment factor for each host in a datacenter and map the VMs according to that factor. Estimated finish time [18] 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.

3 Modified Bee Colony Algorithm for Load Balancing

The proposed method uses the foraging behaviour of honey bees 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.

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 a nectar and bees are ready to take off from the food source.

The architecture for the proposed load balancing method is given in Fig. 1. 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 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.

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 proposed load balancing and scheduling mechanism works in four different steps. 1. VM Current Load Calculation. 2. Load Balancing and Scheduling Decision 3. VM Grouping 4. Task Scheduling.

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

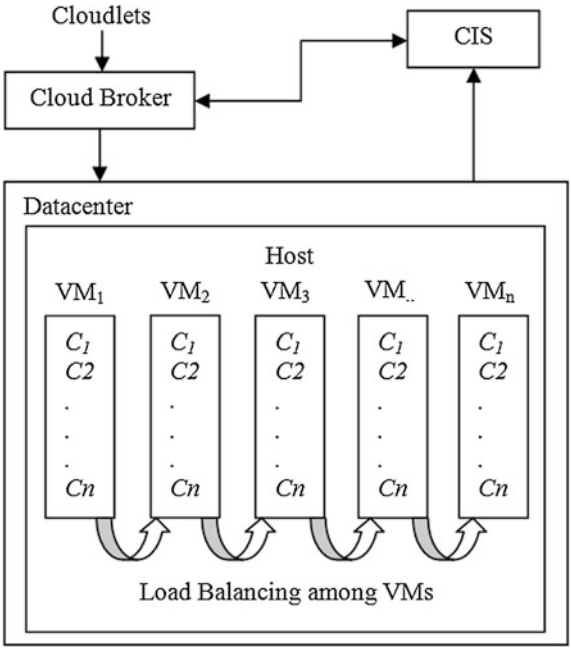

Fig. 1 Load balancing architecture

3.1 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 Eq. (1) the current load can be calculated.

$$Load_{VM} = \frac{N*Len}{MIPS} \tag{1}$$

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 Eq. (2).

$$Load_{DC} = \sum_{vm=1}^{n} Load_{VM} \tag{2}$$

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

$$Capacity_{VM} = PE_{num} * PE_{mips} + VM_{bw} \tag{3}$$

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 Eq. (4),

$$Capacity_{DC} = \sum_{vm=1}^{n} Capacity_{VM} \tag{4}$$

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

$$PT = \frac{CurrentLoad}{Capacity} \tag{5}$$

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

$$PT_{DC} = \frac{Load_{DC}}{Capacity_{DC}} \tag{6}$$

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

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

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

3.2 Load Balancing and Scheduling Decision

In this phase load balancing and rescheduling of tasks are decided. This decision depends on the SD value calculated using Eq. (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.

3.3 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.

3.4 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 Eq. (8),

$$Supply_{VM} = Capacity - Load \tag{8}$$

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

$$Demand_{VM} = Load - Capacity \tag{9}$$

The enhanced bee colony algorithm was given in Fig. 2. 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.

4 Experimental Results

The performance analysis of the proposed method is carried out in a simulated environment. In this heterogeneous environment VMs having different specification are considered. Cloudlets with varying specifications are submitted into this cloud

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

Fig. 2 Enhanced bee colony based load balancing algorithm

environment. The number of migrations and makespan are measured and compared with existing methods.

The makespan time of the proposed method with bee colony algorithm is shown in Table 2. The overall task completion time, i.e. makespan is graphically represented in Fig. 3. 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.

If number of task migrations is greater it will adversely affects the performance of the cloud and thereby reduces the QoS. A good load balancing and scheduling mechanism will reduce the number of task migrations. The proposed method is analyzed for number of task migrations. The results are tabulated in the Table 3.

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.

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

Table 2 Comparison of makespan

Number of cloudlets	Bee colony (s)	Enhanced bee colony (s)
10	50.1	43.85
15	70.1	68.85
20	80.1	78.85
25	110.1	100.1
30	120.1	118.85

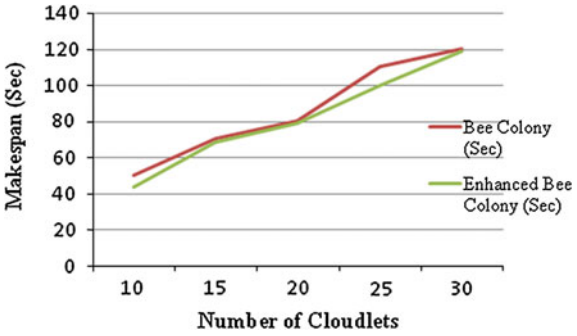

Fig. 3 Comparison of makespan

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

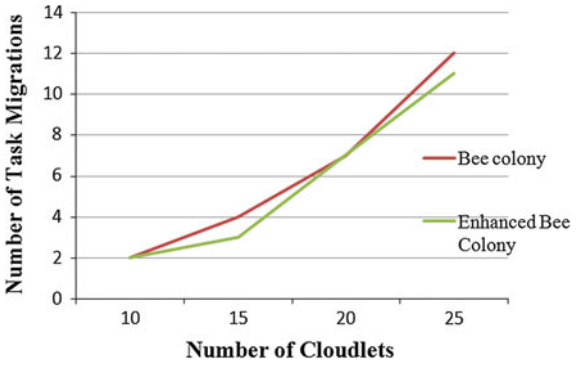

Fig. 4 Number of task migrations

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. 4.

5 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 removes the tasks from overloaded VMs and submitted it 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. Karaboga, D., Basturk, B.: A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. Springer, J. Glob. Optim. **39**, 459–471 (2007)
2. Ajit, M., Vidya, G.: VM level load balancing in cloud environment.: In: IEEE Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT), pp. 1–5 (2013)
3. 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: 2nd World Conference on Complex Systems (WCCS), pp. 594–598 (2014)
4. Remesh Babu, K.R., Mathiyalagan, P., Sivanandam, S.N.: Pareto-Pareto based hybrid Meta heuristic ABC—ACO approach for task scheduling in computational grids. Int. J. Hybrid Intell. Syst. **11**(4/2014), 241–255 (2014)
5. Madivi, R., Kamath, S.S.: An hybrid bio-inspired task scheduling algorithm in cloud environment. In: International Conference on Computing, Communication and Networking Technologies (ICCCNT), pp. 1 –7 (2014)
6. Wang, L., Zhou, G., Xu, Y., Liu, M.: An enhanced Pareto-based artificial bee colony algorithm for the multi-objective flexible job-shop scheduling. Int. J. Adv. Manuf. Technol. **60** (Issue 9–12), 1111–1123. Springer (2012)
7. Domanal, S.G.R., Ram Mohana, G.: Load balancing in cloud computing using modified throttled algorithm. In: IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), pp. 1–5 (2013)
8. Shridhar, G.D., Reddy, G.R.M.: Optimal load balancing in cloud computing by efficient utilization of virtual machines. In: IEEE Sixth International Conference on Communication Systems and Networks (COMSNETS), pp. 1–4 (2014)
9. Sharma, A., Peddoju, S.K.: Response time based load balancing in cloud computing. In: International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), pp. 1287–1293 (2014)
10. Soni, G., Kalra, M.: A novel approach for load balancing in cloud data center. In: IEEE International Conference on Advance Computing Conference (IACC), pp. 807–812 (2014)
11. Dam, S., Mandal, G., Dasgupta, K., Dutta, P.: An ant colony based load balancing strategy in cloud computing. Springer Advanced Computing, Networking and Informatics, Vol. 28, pp. 403–413 (2014)
12. Mohammadreza, M., Amir, M.R., Anthony, T.C.: Cloud light weight: a new solution for load balancing in cloud computing. In: International Conference on Data Science and Engineering (ICDSE), pp. 44–50 (2014)
13. Chen, L., Shen, H., Sapra, K.: RIAL: resource intensity aware load balancing in clouds. In: IEEE Conference on Computer Communications (INFOCOM), pp. 1294–1302 (2014)

14. Xu, G., Pang, J., Fu, X.: A load balancing model based on cloud partitioning for the public cloud. In: IEEE Journal of Tsinghua Science and Technology, pp. 34–39 (2013)
15. Randles, M., Lamb, D., Taleb-Bendiab, A.: A comparative study into distributed load balancing algorithms for cloud computing. In: Proceedings of the IEEE 24th International Conference on Advanced Information Networking and Applications, Perth, Australia, pp. 551–556 (2010)
16. Yao, J., He, J.: Load balancing strategy of cloud computing based on artificial bee algorithm. In: IEEE 8th International Conference on Computing Technology and Information Management (ICCM), pp. 185–189 (2012)
17. Samal, P.: Analysis of variants in Round Robin Algorithms for load balancing in cloud computing. Int. J. Comput. Sci. Inf. Technol. **4**(3), 416–419 (2013)
18. Remesh Babu, K.R., Samuel, P.: Virtual machine placement for improved quality in IaaS cloud. In: IEEE Fourth International Conference on Advances in Computing and Communications (ICACC), pp. 190–194 (2014)