Journal of Ambient Intelligence and Humanized Computing (2024) 15:973–981
https://doi.org/10.1007/s12652-018-1106-7

ORIGINAL RESEARCH

Analysis of load balancing in cloud data centers

Sweekriti M. Shetty[1] · **Sudheer Shetty**[1]

Received: 17 March 2018 / Accepted: 21 October 2018 / Published online: 7 January 2019

Abstract
Cloud computing is a distributed computing system, where the user will utilize the dynamically provisioned resources including storage, processing, network, etc. This has given rise to cloud data centers, which constitutes virtual resources, that will be shared among multiple users. The major issue in cloud data centers is to handle the millions of simultaneous requests/loads from users. To handle such requests efficiently load balancing algorithms are devised. The incoming load has to be distributed fairly and consistently among the machines which are available. Thus, load balancing techniques deals in achieving high resource utilization by sharing the load efficiently. In this work, Modified Central Load Balancer (MCLB) algorithm is proposed, where the load is balanced among all the available virtual machines thereby avoiding overloading and under loading of virtual machines. Allocation of jobs is done by considering the priority and the state of the virtual machine which helps in the fair allocation of the jobs and efficient user utilization. The MCLB algorithm is simulated using CloudSim and it is compared with existing Round Robin algorithm, Throttled algorithm and Equally Spread Current Execution Load algorithm. The comparison analysis shows that MCLB outperforms the remaining in performance evaluation metrics such as response time, data center processing time and total cost.

1 Introduction

The area of distributed computing has evolved over the years with advancements in network technologies. Cloud Computing has been widely used to refer to different technologies, services, and concepts. It is often associated with virtualized infrastructure or hardware on demand, utility computing, IT outsourcing, platform and software as a service, and many other things that now are the focus of the IT industry (Rodriguez and Buyya 2014). It aims at allowing user to avail the benefits of existing technologies, without having deep knowledge about them. It offers three different kinds of service, namely:

- Software as a Service (SaaS): SaaS provides applications and services to the users based on their demands. SaaS generally consists of desktop applications such as office automation, photo and video editors, Enterprise Resource Planning softwares and so on which are running on the providers infrastructure and made accessible through a web browser at the user's end when user demands it.
- Platform as a Service (Paas): PaaS is responsible for providing scalable and elastic runtime environments for the user on demand. It also lets the user to host the execution of applications at its end. The services provided by PaaS are aided by middleware platform, which creates an abstract environment, where the user applications are hosted and run. The service provider of PaaS must make sure that scalability is achieved when user demands are more and to manage fault tolerance. The end users will create applications making use of service providers APIs and system libraries.
- Infrastructure as a Service (Iaas): IaaS provides infrastructure such as virtual hardware, storage and networking on demand to the end users. Using IaaS all the end user computations can be achieved remotely using virtual hardware by utilizing the VM instances.

Virtualization is the core technology in cloud computing, mainly in infrastructure based services. Virtualization is used to create an customizable execution environment for the applications. Virtualization is realized by using software or combination of software and hardware to emulate

✉ Sweekriti M. Shetty
shettysweekrithi86@gmail.com

Sudheer Shetty
sudheer.cs@sahyadri.edu.in

[1] Sahyadri College of Engineering and Management, Adyar, India

an execution environment which is clearly different than the host which creates this application (Sanaei et al. 2014). The main components in Virtualization are: the guest, the host, and the virtualization layer.

- The guest represents the system component that interacts with the virtualization layer.
- The host represents the original environment where the guest is supposed to be managed.
- The virtualization layer is responsible for recreating the same or a different environment where the guest will operate.

In the case of hardware virtualization, the guest is represented by a system image comprising an operating system and installed applications. These are installed on top of virtual hardware that is controlled and managed by the virtualization layer, also called the VM manager. The host is instead represented by the physical hardware, and in some cases the operating system, that defines the environment where the VM manager is running. The main common characteristic of all these different implementations is the fact that the virtual environment is created by means of a software program. The ability to use software to emulate such a wide variety of environments creates a lot of opportunities.

The applications of Cloud Computing are in diversified fields of data analytics, machine learning and deep learning and so on (Tokle et al. 2014; Fernandes et al. 2017). Thus it is of high importance that cloud computing technologies and its associated algorithms are efficient.

Cloud Computing is an heterogeneous computing technology, which is implemented at the data centers. The data centers heterogeneity is present at various levels such as instruction level architecture, operating system, application level and so on (Zhang et al. 2014). This however poses certain challenges to the data centers which will be dealt in this paper. The major challenges posed to the data centers are (Liang and Yang 2013):

1. The resources in the data centers have to be systematically provisioned in order to handle the incoming service requests.
2. The efficient allocation of the various tasks to the available resources so that the load is balanced across all the resources worldwide.

If the above challenges are overcome, then the Quality of Service (QoS) provided by service provider will increase, thereby confirming that there is no single point of failure. The main challanges faced by load balancing algorithms are (Al Nuaimi et al. 2012):

1. Distribution of cloud data center: the cloud data center servers are spatially distributed all over the globe. Most algorithms designed works fine for servers within a grid and located close by, where the delay in data communication is minimal or negligible. Thus, there is a need to develop efficient load balancing algorithms for data center servers which are distance apart connected via networks that are prone to network delays.
2. Storage and replication: data centers replicate the data to have a backup such that data is available all the time including any major catastrophic failures. In case of replicating data fully into the servers will incur more cost as the amount of storage required will be quite large. Thus, partial replication of data is done at different data center severs based on its processing power and available storage. This approach will lead to better utilization of available severs but will increase the complexity of load balancing algorithms due to the fact that replication is done partly across multiple severs.
3. Complexity of the algorithm: load balancing algorithms are preferred to have less execution time because they need to run across multiple cloud servers. Complex algorithms with higher execution time will generally lead to delay thereby reducing the efficiency drastically.

Load balancing algorithms are classified mainly into two categories:

1. Dynamic load balancing algorithm: in this category of algorithms, load is allotted to a VM based on the current state of the machine i.e during the rum time (Wang et al. 2014). State of the VM can be considered with respect to various characteristics like available memory, CPU speed, etc. Dynamic algorithms can be further classified into: off-line mode and online mode. Offline mode algorithms are those where the allotment of task to the VMs are done only at predefined time intervals. Whereas in online mode, immediate allotment of the task is done.
2. Static load balancing algorithm: in these category of algorithms, the VM information like capacity, memory size, performance, etc is known in advance and these details are considered for all the allocations. The change in these information during the run time do not affect the allotment of the task to VMs. Static algorithms are easy to incorporate but is not an good option for heterogeneous cloud environment (Liang and Yang 2013).

The load balancing algorithms are typically evaluated through the following performance evaluation metrics (Randles et al. 2010):

- Reliability: if the system performs consistently as per the specification, then the system is reliable. In case of

system failure, the task is performed by other VM to increase stability of the system.
- Accuracy: it is the measure of difference between the expected result and the obtained result. As the difference decreases the accuracy increases.
- Throughput: it is the measure of system performance where the number of instructions executed per unit time is considered. In load balancing, throughput specifies the number of tasks executed by the VM per unit time.
- Scalability: capacity of the system to perform under unacceptable situations can be measured using scalability. In load balancing system, scalability is surviving when the task size or workload increases by incorporating appropriate mechanism.
- Makespan: time taken to complete all tasks submitted to the system can be referred to as makespan. If proper load balancing techniques are employed, then the results will be in optimal makespan.

2 Related work

Over the years several cloud load balancing algorithms were designed and developed employing various techniques. A biologically inspired bee colony optimization technique was developed by Nakrani and Tovey (2004), to load balance the distributed cloud hosting web services. In this technique, when the number of requests for web services varies, the allocation of web servers varies accordingly. Here, the servers are deemed as virtual servers which contains a service request queue. A measure called "profit" is calculated by the server which gives the service of the application request, based on performance metric like CPU time, etc. There is also "advert board" which indicates the idle servers whether any services are needed by them. The overall profit of the corresponding infrastructure is calculated based on different metrics. Here, the severs will play the role of bees and idle servers will be like the bees waiting to fetch the nectar. Another biologically inspired load balancing technique based on the modified ant colony optimization (ACO) technique was developed by Nishant et al. (2012). ACO is based on making use of movement of ants as a phenomena in a certain direction. This is related to cloud load balancing by considering software module as ants and the VM's as nodes. Initially, one node will be selected as the Regional Load Balancing Node (RLBN), which acts as the head node. Ants will originate from this RLBN and will be able to move in either direction towards the nodes, based on the load of nodes. In this approach a pheromone table is maintained, which maintains information about loads on the nodes. The main aim of ant is to find the least loaded node and assign application to it, thereby equally distributing the node in the network.

A software module called load balancer has been proposed by Jain et al. (2013). This load balancer is used to allocate VMs to multiple user application requests, based on the availability of VMs, such that the allocation strategy becomes optimal. The load balancer will maintain a queue for the user application request and then suitably assign the appropriate VM. It also maintains the information regarding allotments to the VMs, thereby knowing in advance which VMs are free. A variation of load balancer called as "active VM load balancer" was developed by Adhikari and Patil (2013). In this approach, the VM which is having less load will be assigned to the new application request. The "active VM load balancer" will send the ID of the VM to the data center controller, which sends requests to the VM having the ID received by it.

Scheduling of the VMs was also an important factor to utilize the resources to maximum extent. Round Robin based load scheduling algorithms had limitation that, once the VM is allocated to a user application request, then its state will not maintained. This drawback leads to the execution of the algorithm once again for the same request. An improved Round Robin based scheduler was proposed by Mahajan et al. (2013), which maintains the state each time an application request is run by the server. This was done by using specific data structures called hash map to keep information about the VM allocated to a specific user application request and VM state list, which maintains the status of the VM (i.e. whether busy or free). Simulation results showed that this algorithm increased the response time as compared to the ordinary Round Robin scheduler. A modified throttled algorithm for load balancing was developed by Domanal and Reddy (2013). In this technique, the best available VM is allocated to an application request based on the response time and processing time. Here "throttled VM load balancer" maintains the status of every VM. Upon an application request to the data center controller, it asks the "throttled VM load balancer" to assign the perfect VM based on the application requirements.

Authors in Xu et al. (2013) have developed a load balancing algorithm by applying the concepts of game theory to partition the cloud so that higher efficiency is obtained. A stochastic model based job scheduling algorithm was developed by Maguluri and Srikant (2014), where the server at the data center chooses the jobs to be scheduled for execution based on availability of resources. The scheduling strategy and resource provisioning for scientific workflow for IaaS cloud was proposed (Rodriguez and Buyya 2014), where the large dataset of workflows were managed efficiently by applying Particle Swarm Optimization (PSO). Here, the PSO techique has been implemented using CloudSim for four different workflows. Authors in Cao et al. (2014) have developed techniques based on queuing model to optimize the performance and reduce the power consumption in data

centers. A heuristic based load balancing algorithm is developed by authors in Zhao et al. (2016), where the clustering approach is used. They have applied Baye's theorem to obtain optimal clusters of physical hosts available for load balancing. The data intensive applications which manage tera bytes and peta bytes of data use cloud based workflow applicaions for processing the data. These workflow based cloud also need to balance their load and manage resources efficiently (Wang et al. 2014; Zhang et al. 2014). The power consumption at cloud data centers is quite huge which gave rise to techniques for optimal power usage in heavily loaded data centers (Cao et al. 2014; Tai et al. 2014; Wang et al. 2014). Resource allocation and scheduling of tasks to various resources in cloud data centers have been extensively studied in literatures (Fang et al. 2010; Maguluri and Srikant 2014; Maguluri et al. 2014). The various literatures with its contributions, limitations is highlighted in Table 1.

3 Proposed methodology and architecture

In the proposed algorithm, all the requests from the users all around the world arrive at the data center controller, which is one of the major component of the cloud computing. The data center controller forwards the requests to the proposed MCLB algorithm to assign the request to the available VMs. This algorithm maintains a table which contains the id's of the VM, priority of the VM and the state of the VM. The algorithm will search the table to find the VM with highest priority and which is available at that moment. If found, the algorithm will reply back to the data center controller with the id of that machine (VM_{id}) and the data center controller will assign the request to that machine, else, it will wait for the VM to be free and once free that machine will be assigned to the request. The data center model of the proposed MCLB algorithm is depicted in Fig. 1.

4 Implementation details

The sequential flow of MCLB is highlighted in Fig. 2. The proposed algorithm mainly consists of following modules:

- User module: this module is used to create user bases. Here you can specify the number of user bases participating in simulation, ID of the user base, name of the user base and other characteristics. The functions performed by this module is sending the request to the data center controller and receiving the response cloudlet from the data center controller.
- Data center controller module: this module does the load balancing with the help of the proposed algorithm. On receiving requests from the user bases, it will use the algorithm to find the VM which is available and has the highest priority. On finding the VM, the job will be assigned to that VM. If all the VMs are busy serving requests, the algorithm will reply back to the data center controller saying that all machines are busy. Now the Data Center Controller will have to wait for the signal from the VM which gets free and then assign the job that is in the queue to that VM. Once the job is assigned, data center controller should update the table which is maintained by the algorithm about the job allocation. The VM will process the job assigned to it and will reply back with the response cloudlet to the data center controller. This response is sent to the user who has sent the request. Also, the table is updated regarding the de-allocation of the VM and the state of that VM will be set to available. This module communicates directly with the other three modules of the system. All the requests from the users arrive at the data center controller, which is then forwarded to the VM by referring the algorithm and the table maintained by the algorithm for processing. The response cloudlet from the VM arrives at data center controller which is then sent to the appropriate users.
- VM Creation module: this module is used to create the VMs. VMs are the machines which are not real but replicate the properties of real system. Such machines are used to efficiently use the available resources and improve the speed of processing requests. VMs can be created at data center as per the requirements using this module. It also allows us to set the characteristics of the VMs such as memory size, speed of the CPU, etc.
- Maintaining table module: this module is used to maintain the table that is used by the proposed algorithm. The table contains id of the VM (VM_{id}), state of the VM and priority of the VM. Data Center Controller should update the table on allocation of every request to the VM and de-allocation from the VM after the processing of job is complete. VM_{id} indicates the unique identification number which is used to identify the VM.

  Priority of the VM is calculated based on the CPU speed and the memory of the VM. The formula is given as:

$$P_r(i) = t \times T_c(i) + s \times T_m(i) \tag{1}$$

Here t represents the CPU Weight i.e time of host CPU that is available for the Virtual CPUs execution.

s represents the memory weight i.e size of memory available for the VM. T_m is the Memory resource available. T_c is the CPU speed (MIPS). $t + s = 1$ i.e t represents the % of CPU time available for a particular VM out of the total availability and s is the % of memory of total memory available for a particular VM.

Table 1 Summary of literature indicating the load balancing algorithms

Literature	Contribution	Limitations/future perspective
Fang et al. (2010)	✓ A two level scheduling algorithm is proposed using the CloudSim framework	✓ Develop precise model based on more user requirements including bandwidth, cost and so on
	✓ Issues of flexibility and virtualization taken into consideration	
Mashaly and Kuhn (2012)	✓ Proposed a laod balancing approach for cloud based content delivery networks	✓ Power consumption is more
	✓ Uses server activation and deactivation technique to transfer extra load to cloud servers which are free	✓ Extend the scheme to more realistic data center architecture
Abdullah and Othman (2013)	✓ Divisible load theory approach is designed for scheduling tasks	✓ Communication overhead, dynamic workloads and real time tasks have to be considered
	✓ Serves with better QoS requirements for user tasks	
Hsiao et al. (2013)	✓ Resource alloacation policies have been designed for virtualized cloud setup	✓ Develop resource allocation model for fine grained time specific workloads
	✓ A framework based on a mixed-integer nonlinear optimization is developed for scalable cloud servers	✓ Make use of green resources in multiple data centers
Carlini et al. (2013)	✓ Distributed Virtual Environment architecture is proposed for cloud data centers	✓ Architecture will be applied to dynamic cloud provisioning
	✓ The computational cost is reduced by employing greedy heuristics	✓ Adopt point to point gossip algorithm for faster access of cloud servers.
Fan et al. (2014)	✓ An improved MapReduce technique is developed for heterogeneous clusters	✓ Load balancing will be applied to the map phase
	✓ Partitioning is employed in Reduce phase to overcome overhead occurring due to skewed data	
Ghafarian and Javadi (2014)	✓ A workflow scheduling approach is proposed for data intensive applications	✓ Provisioning policy that uses more information about volunteer resources
	✓ High QoS constraints are met for scientific workflows	
Hsiao et al. (2013)	✓ Using Hadoop based system a distributed load balancing algorithm	✓ Error propagation due to measurement noise in distance and angle will increase in the transformation matrices
	✓ Reduces the network traffic laod on cloud servers thereby increasing the bandwidth	✓ Authors plan to adopt cluster-based paradigm in future design
Luo et al. (2014)	✓ A research on energy minimization at Internet Data Center (IDC) is carried out	✓ Employing in real time Internet Data Centers
	✓ The research proposes the eco-IDC algorithm for IDC workload scheduling which reduces the energy cost	✓ Authors plan to adopt cluster-based paradigm in future design
Cao et al. (2014)	✓ An optimization problem is solved for power allocation and load distribution in multiple cloud servers	✓ Apply the strategy for other server systems and distributed computing
	✓ A multi-variable optimizer is developed by modelling mutiple servers in queuing model	
	✓ Increases power management and resource management at cloud data centers	
Wang et al. (2014)	✓ A workload balancing and resource management algorithm is proposed for SWIFT storage systems in cloud environment	✓ Employing in real time Internet Data Centers
	✓ Workload balance is done by making use of split, merge and pair algorithms	✓ Authors plan to adopt cluster-based paradigm in future design
Khara and Thakkar (2017)	✓ A research on energy minimization at Internet Data Center (IDC) is carried out	✓ Employing in real time Internet Data Centers
	✓ The research proposes the eco IDC algorithm for IDC workload scheduling which reduces the energy cost	✓ Authors plan to adopt cluster-based paradigm in future design

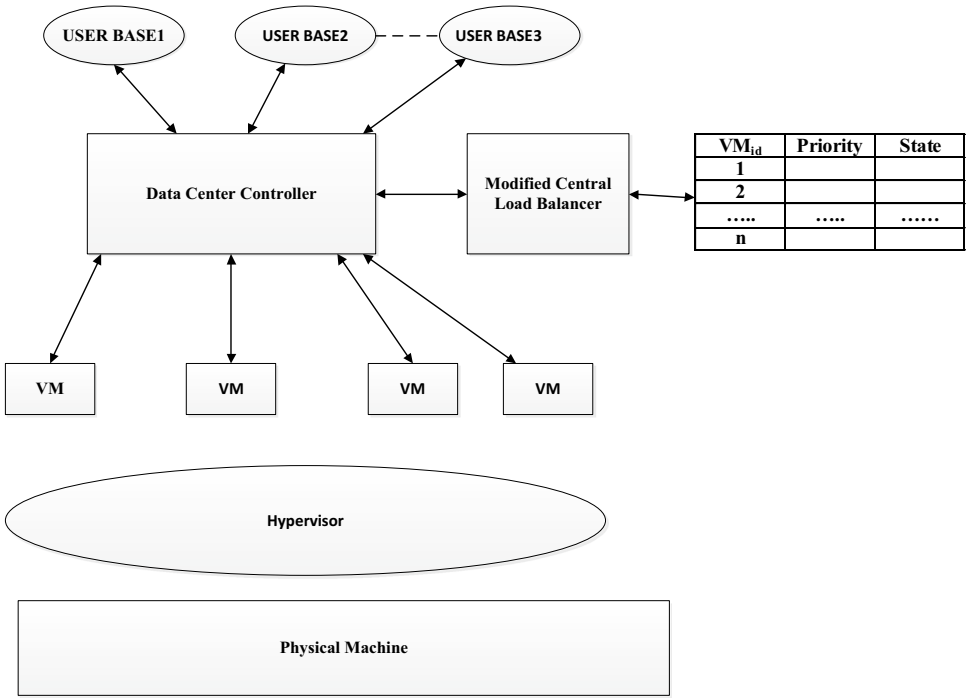

Fig. 1 Data center model for the proposed MCLB algorithm

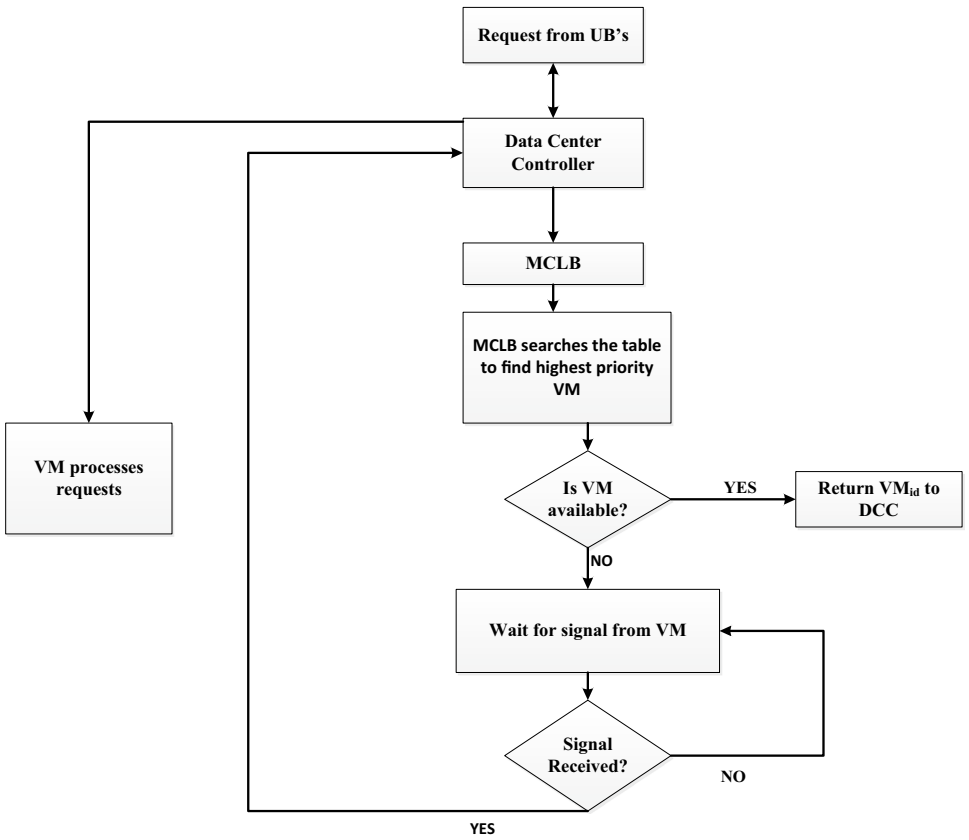

Fig. 2 Flowchart of MCLB algorithm

Algorithm 1 The MCLB algorithm

```
Input: Request for VM
Output: ID of VM
while There exists request for VM do
    for Every VM do
        Check the state of VM
        if VM is free then
            Calculate priority as per equation 1
        else
            Set priority to 0
        end if
        Assign priority
    end for
end while
```

The proposed MCLB algorithm is described in Algorithm 1.

5 Results and analysis

The MCLB algorithm is simulated using the CloudSim framework, which is used for simulating cloud computing infrastructure and services (Calheiros et al. 2011). The MCLB algorithm is compared with the existing algorithms and is graphically analyzed considering various parameters stated in Sect. 1. The various compared algorithms and their conventions are:

- Round Robin Algorithm (RRA)
- Throttled Algorithm (TA)
- Equally Spread Current Execution Load Algorithm (ESLBA)

5.1 Main configuration

As we can see in the Fig. 3, six user bases are created in six different regions. Requests per user per hour and the size of the data per request are kept same for all the user bases to obtain proper result. Different peak hours are set for different user bases and different average peak users and off peak users are set as shown in Fig. 3. Service Broker policy is set to Closest Data Center where the data centers receive the traffic from the user bases that are located near to that data center. Since we are using only one data center, all the user bases will send their requests only to that data center.

One Data center is created with 50 VMs. The configurations of the data center is as shown in Fig. 4 . Here, the region where the data center should be located in, operating system, costs etc are shown.

Figure 5 shows the advanced configuration, where the number of users forming one user base, request grouping factor and length of the instruction per request are set. All the configurations shown above remains the same for every simulation, but the load balancing policies are changed for every simulation and values are taken down for comparison and to plot graphs.

5.2 Evaluation parameters

1. Response time: for the configurations set as discussed above, we have considered six user bases to estimate the response time by the region for various algorithms. is depicted in the graph obtained as shown in . It is observed from the graph in Fig. 6 that the proposed MCLB algorithm has the minimum response time compared to the other algorithms for every user base. This

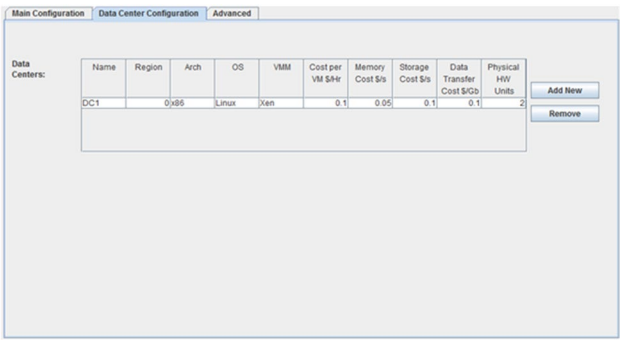

Fig. 4 Snapshot of data center configuration

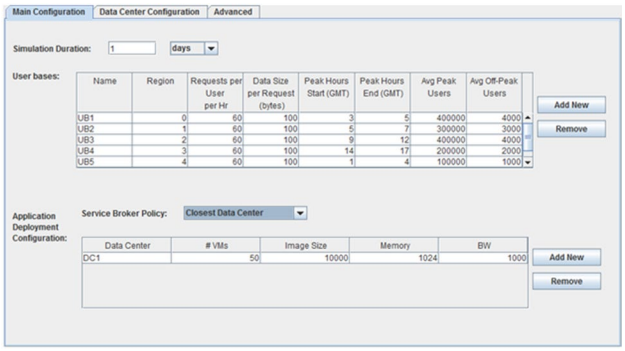

Fig. 3 Snapshot of main configuration

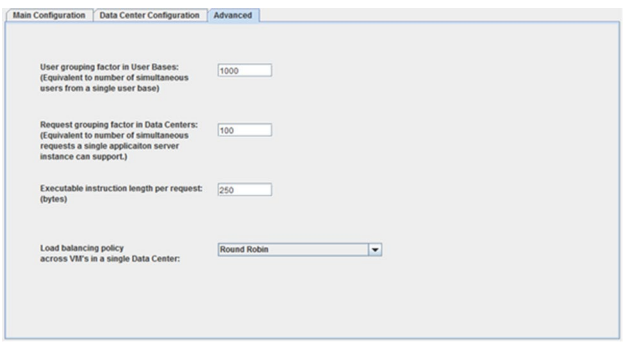

Fig. 5 Snapshot of advanced configuration

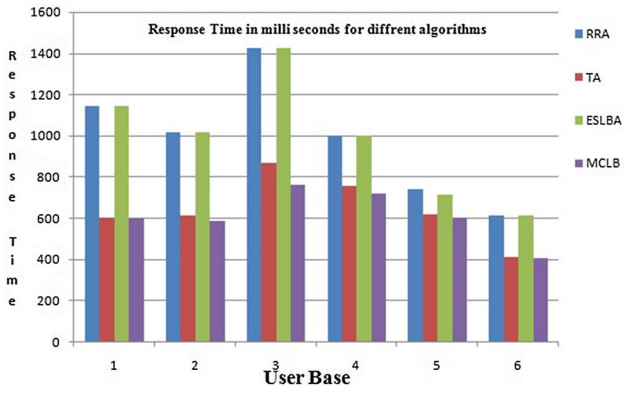

Fig. 6 Comparison of response time for various algorithms

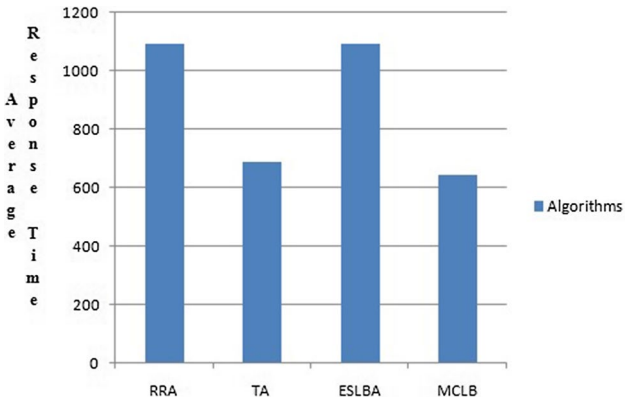

Fig. 7 Comparison of average response time for various algorithms

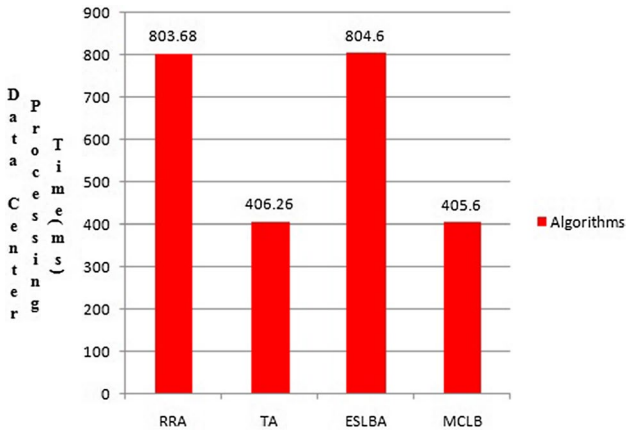

Fig. 8 Comparison of data center processing time

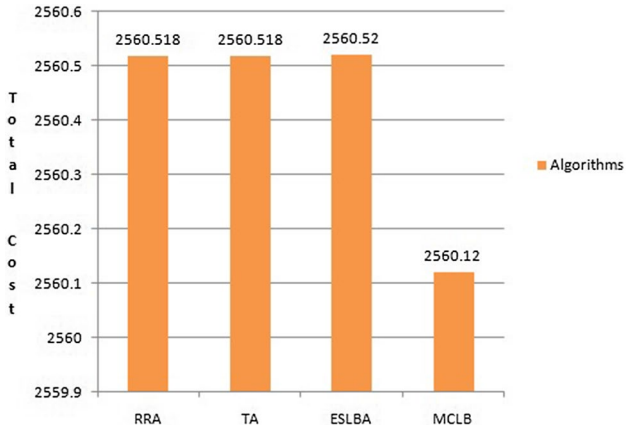

Fig. 9 Comparison of total cost

is due to the fact that MCLB takes up task based on priority and looks for VMs which are lightly loaded.

2. Average response time: for the configurations set as discussed above, the average response time for various algorithms is depicted in the graph obtained as shown in Fig. 7. From the graph we can conclude that the average response time for the user requests is less in the proposed MCLB algorithm as compared to other algorithms when distributing the load among available VMs.
3. Data center processing time: data center processing time is the time required to process the requests of user at the data center. Data center has to use the algorithm and forward the request to the appropriate VMs. The time taken for this process is data center processing time. For the configurations set as discussed above, the data center processing time for various algorithms is depicted in the graph obtained as shown in Fig. 8. It is observed that RRA and ESLBA takes almost double the time taken by TA and MCLB to process the request at Data Center. MCLB take slight lesser time than TA and hence is efficient compared to other algorithms.
4. Cost: the total cost includes cost per VM, storage cost, memory cost and data transfer cost. For the configurations set as discussed above, the total cost for various algorithms is depicted in the graph obtained as shown in Fig 9. Since the configurations set for comparing all the algorithms are same, there is no much difference in the total cost. However, MCLB reduces the overall cost compared to other algorithms.

6 Conclusion and future work

In this work we have proposed an MCLB algorithm. The MCLB algorithm balances the load among all the available VMs and thus takes care of the overloading and under loading of VMs. Allocation of the jobs is done by considering the priority and the state of the VM which helps in the

fair allocation of the jobs and efficient resource utilization. Simulation results have a shown that, proposed MCLB is more efficient compared to Round Robin Algorithm, Throttled Algorithm and Equally Spread Current Execution load Algorithm in terms of performance evaluation metrics such as response time, average response time, data center processing time amd cost.

As part of future work we can consider live migration of VMs and more sophisticated auto scaling approaches. Also a load balancing algorithm can be developed by considering the processor utilization and memory utilization.

References

Abdullah M, Othman M (2013) Cost-based multi-qos job scheduling using divisible load theory in cloud computing. Procedia Comput Sci 18:928–935

Adhikari J, Patil S (2013) Double threshold energy aware load balancing in cloud computing. In 2013 fourth international conference on computing, communications and networking technologies (ICCCNT) (pp 1–6)

Al Nuaimi K, Mohamed N, Al Nuaimi M, Al-Jaroodi J (2012) A survey of load balancing in cloud computing: challenges and algorithms. In: Network cloud computing and applications (NCCA), 2012 second symposium on (pp 137–142)

Calheiros RN, Ranjan R, Beloglazov A, De Rose CA, Buyya R (2011) Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Softwa Pract Exp 41(1):23–50

Cao J, Li K, Stojmenovic I (2014) Optimal power allocation and load distribution for multiple heterogeneous multicore server processors across clouds and data centers. IEEE Trans Comput 63(1):45–58

Carlini E, Ricci L, Coppola M (2013) Flexible load distribution for hybrid distributed virtual environments. Future Gener Comput Syst 29(6):1561–1572

Domanal SG, Reddy GRM (2013) Load balancing in cloud computing using modified throttled algorithm. In: Cloud computing in emerging markets (CCEM), 2013 IEEE international conference on (pp 1–5)

Fan Y, Wu W, Xu Y, Chen H (2014) Improving mapreduce performance by balancing skewed loads. Commun China 11(8):85–108

Fang Y, Wang F, Ge J (2010) A task scheduling algorithm based on load balancing in cloud computing. In: Web information systems and mining. Springer, pp 271–277

Fernandes SL, Gurupur VP, Sunder NR, Arunkumar N, Kadry S (2017) A novel nonintrusive decision support approach for heart rate measurement. Pattern Recogn Lett

Ghafarian T, Javadi B (2014) Cloud-aware data intensive work flow scheduling on volunteer computing systems. Future Gener Comput Syst 51:87

Hsiao H-C, Chung H-Y, Shen H, Chao Y-C (2013) Load rebalancing for distributed file systems in clouds. Parallel Distrib Syst IEEE Trans 24(5):951–962

Jain A, Yadav A, Namboodiri L, Abraham J (2013) A threshold band based model for automatic load balancing in cloud environment. In: Cloud computing in emerging markets (CCEM), 2013 IEEE international conference on, pp 1–7

Khara S, Thakkar U (2017) A novel approach for enhancing selection of load balancing algorithms dynamically in cloud computing. In: Computer, communications and electronics (comptelix), 2017 international conference on, pp 44–48

Liang P-H, Yang J-M (2013) Evaluation of cloud hybrid load balancer (CHLB). Int J E-Bus Dev 41:23

Luo J, Rao L, Liu X (2014) Temporal load balancing with service delay guarantees for data center energy cost optimization. Parallel Distrib Syst IEEE Trans 25(3):775–784

Maguluri ST, Srikant R (2014) Scheduling jobs with unknown duration in clouds. IEEE/ACM Trans Netw 22(6):1938–1951

Maguluri ST, Srikant R, Ying L (2014) Heavy traffic optimal resource allocation algorithms for cloud computing clusters. Perform Eval 81:20–39

Mahajan K, Makroo A, Dahiya D (2013) Round robin with server affinity: a VM load balancing algorithm for cloud based infrastructure. J Inf Process Syst 9(3):379–394

Mashaly M, Kuhn PJ (2012) Load balancing in cloud-based content delivery networks using adaptive server activation/deactivation. In: Engineering and technology (ICET), 2012 international conference on, pp 1–6

Nakrani S, Tovey C (2004) On honey bees and dynamic server allocation in internet hosting centers. Adapt Behav 12(3–4):223–240

Nishant K, Sharma P, Krishna V, Gupta C, Singh KP, Rastogi R et al (2012) Load balancing of nodes in cloud using ant colony optimization. In: 2012 14th international conference on modelling and simulation, pp 3–8

Randles M, Lamb D, Taleb-Bendiab A (2010) A comparative study into distributed load balancing algorithms for cloud computing. In: Advanced information networking and applications workshops (WAINA), 2010 IEEE 24th international conference on, pp 551–556

Rodriguez MA, Buyya R (2014) Deadline based resource provisioning and scheduling algorithm for scientific workflows on clouds. IEEE Trans Cloud Comput 2(2):222–235

Sanaei Z, Abolfazli S, Gani A, Buyya R (2014) Heterogeneity in mobile cloud computing: taxonomy and open challenges. IEEE Commun Surv Tutor 16(1):369–392

Tai J, Li Z, Chen J, Mi N (2014) Load balancing for cluster systems under heavy-tailed and temporal dependent workloads. Simul Modell Pract Theory 44:63–77

Tokle S, Bellipady SR, Ranjan R, Varma S (2014) Energy-efficient wireless sensor networks using learning techniques. Achievements and trends, case studies in intelligent computing, pp 407–426

Wang Z, Chen H, Fu Y, Liu D, Ban Y (2014) Workload balancing and adaptive resource management for the swift storage system on cloud. Future Gener Comput Syst 51:120

Xu G, Pang J, Fu X (2013) A load balancing model based on cloud partitioning for the public cloud. Tsinghua Sci Technol 18(1):34–39

Zhang H, Jiang G, Yoshihira K, Chen H (2014) Proactive workload management in hybrid cloud computing. Netw Serv Manag IEEE Trans 11(1):90–100

Zhao J, Yang K, Wei X, Ding Y, Hu L, Xu G (2016) A heuristic clustering-based task deployment approach for load balancing using bayes theorem in cloud environment. IEEE Trans Parallel Distrib Syst 27(2):305–316