Indian Journal of Science and Technology, Vol 10(11), DOI: 10.17485/ijst/2017/v10i11/110107, March 2017
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

An Efficient Load Balancing Scheme for Cloud Computing

Atyaf Dhari* and Khaldun I. Arif

Department of Computer Science, College of Education for Pure Science, Thi_Qar University, Iraq;
Atyafcomsinc@gmail.com,khaldun.i.a.2014@gmail.com

Abstract

Objectives: The load balancing becomes an important point for performance and stability of the system. Therefore, it is needed an algorithm for enhancing the system performance by balancing workload among VMs. **Methods:** Task scheduling algorithms are used to achieve the load balancing and QoS. The proposed Load Balancing Decision Algorithm(LBDA) to manage and balance the load between the virtual machines in a datacenter along with reducing the completion time (Makespan) and Response time. **Findings:** The mechanism of LBDA is based on three stages, first calculates the VM capacity and VM load to categorize the VMs' states (Under loaded VM, Balanced VM, High Balance VM, Overloaded). Second, calculate the time required to execute the task in each VM. Finally, makes a decision to distribute the tasks among the VMs based on VM state and task time required. **Improvements**: We compared the result of our proposed LBDA with Max-Min, Shortest Job Firstand Round Robin. The results showed that the proposed LBDA is more efficient than the existing algorithms.

Keywords: Cloud Computing, LBDA, Load Balancing, Makespan, Response Time, Task Scheduling

1. Introduction

Cloud computing environment is a computing paradigm for managing and accessing services over the internet it is defined as "a paradigm for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction"[1]. Cloud computing indicates to the applications and the services executing on the deployment network using virtualized resources and restore by common Internet protocols and networking standards. It is developed from the grid computing, distributed computing and parallel computing. Parallel and distributed system included a set of interconnected and virtual computers that are provisioning dynamically[2]. The cloud computing characteristics are basically as on-demand-self-service, broad network access, resource pooling, rapid elasticity, measured service and location dependency and security[3]. Cloud computing provides three kinds of service to consumer i.e., IaaS, PaaS and SaaS[4].

Providing the virtualization technology in cloud platforms helps enterprises to rent computing power in the form of virtual machines to the users. The users may utilize hundreds or thousands of Virtual Machines (VMs)[5]. Even though the cloud has greatly simplified the capacity provisioning process, it poses several novel challenges in the area of Quality of Service (QoS) management. QoS is fundamental for cloud users, who expect providers to deliver the advertised quality characteristics, and for cloud providers, who require to obtaining the right tradeoffs between QoS levels and operational costs[6]. One of important challenges in cloud computing is a load balancing in data centre[7]. To handle a very large size of data many techniques to optimize load and simplify operations are needed to obtaindesirable performance level for the users[8]. There is a need an effective algorithm in term of load balancing in the cloud computing. The load can be differentiated in various categories, CPU load, delay

*Author for correspondence

or network load, memory capacity[5]. Load balancing guarantee that each processor in the system does approximately the same quantity of load at any point of time[8]. The task scheduling is used to determine the suitable resources to execute the tasks that received from users[9]. For increasing the utilization of resources, we use the load balancing algorithm. Efficient utilization is achieved by using the idle resources until finishing the resources of processors that have the high load. The load balancing mechanism divides the loads within all the resources which are available[10]. The dynamic cloud computing used many of load balancing strategies[11]. The Max-Min algorithm for each task, the minimum completion time is calculated, then the task with the maximum of minimum completion time is mapped to the corresponding virtual machine[12]. The Shortest-Job-First (SJF) is a various algorithm to CPU scheduling. This way associated with every process length's next CPU burst. The process with shortest length assigned to CPU when it is available. When there are two processes that have same CPU bursts then scheduling based on FCFS[13]. In Round Robin (RR) approach, the processes are separated between all processor. Every process is mapping the processor in a round robin order by using quantum time. The allocation of process is not based on the allocations from remote processors. The processing time of task for various processes is not equal but the distributions of workload are same[14].

As mentioned earliest, the users send the requests to the service provider, the service provider has to serve many users to provide the best results and take into account maximizing resources utilization. Distributing the tasks into the resources to avoid overloading where there are under loaded resources. The objectives of this work to achieve users' satisfaction and provider satisfaction by balancing the loads and increase resources utilization, by minimizing the average of both makespan time and response time using LBDA algorithm effective performance.

The paper is divided into many Sections. In Section 2, the proposed algorithm is discussed. The experimental result of proposed scheme in Section 3. The performance evaluation is shown in Section 4. The conclusion is provided in Section 5.

Some of studies have been carried for improving load balancing such as:

In [15]used Genetic Algorithm (GA) for ACO (Ant colony optimization) initialization. Second, ACO could arrive at local optimal point and the convergence speed is typically low. In this time, introduce the idea of Simulated Annealing (SA) to avoid a local optimal and accelerate the convergence. The experiments result enhanced ACO achieves good performance in load balancing. Proposed in [1]enhanced weighted round robin algorithm taking into account the capabilities of each virtual machine and the length of each task, where each requested task assigns to the most suitable virtual machines. The experiment results and performance analysis of this algorithm proved that the improved weighted round robin algorithm is the most suitable one where heterogeneous/homogeneous tasks with heterogeneous resources (virtual machines) compared to the other round robin and weighted round robin algorithms.

Author in[9]proposed a new task scheduling strategy based on the total order of resource allocation to enhance the Min-Min algorithm. In term decreasing the total executing time maximizing resources utilization. The experimental results showed that the proposed heuristic algorithm achieves the best values of completion time compared to Sufferage and Min-Min algorithms.

Proposed[16] model that is based on centralized load balancing strategy. In this strategy, the load balancer must bind the tasks to VMs to reduce the response time and turnaround time. Before binding tasks to the virtual machine, first, the load balancer calculates the remaining capacities of all virtual machines and then send the task to the most powerful virtual machine. The experiment result is showed a better performance in terms of average response time and turnaround time compared with Round Robin.

Proposed honey bee behavior inspired load balancing[17] (HBB-LB) to achieve a good balance load through VMs for maximizing the throughput. Along with balancing the priorities of tasks on the VMs in such a way that the amount of waiting time of the tasks in the queue is minimal. The author compared the proposed algorithm with existing load balancing and scheduling algorithms. The experimental results indicate that the proposed algorithm is more efficient.

Author in[18] proposed a new task scheduling algorithm to minimize the makespan and maximize the resources utilization taking into account independent tasks. The proposed algorithm calculates the total processing power of the available resources and the total requested processing power of the users' tasks. Then allocate a group of users' tasks to each virtual machine based on the ratio of its needed power corresponding to the total processing power of all virtual machines. Evaluate the performance

of the proposed algorithm, by comparing the proposed algorithm to the GA, and PSO algorithms. The experimental results indicated that the proposed algorithm outperforms of other algorithms by minimizing make span and maximizing the resources utilization.

In presented[19]task scheduling strategy based on Quantum Particles Swarm algorithm (BLQPSO) to reduce the makespan of execution scientific application workload in cloud computing environments.

Author in[20]proposed task scheduling algorithm to the independent tasks in the cloud computing. The proposed algorithm is a blend of the Cuckoo search (CS) algorithm and PSO algorithm, called PSOCS. To reduce the make span and maximize the utilization ratio.

2. The Proposed Algorithm

In cloud computing environment there are many data centres that composed of heterogeneous resources which consist of servers and virtual machines (VMs). The servers and VMs may have different configurations such as memory sizes, bandwidth, storage and processing capacities.

2.1 Configurations

We have set of virtual machines as VM {VM_1, VM_2, VM_3,…VM_m}. Each VM has different parameters such as VM state and the speed in Million Instructions Per Second (MIPS). All VMs are no interrupted, non-preemptive and independent. The tasks are independent {T_1, T_2, T_3,… T_n}. Each task has different properties such as id, length, start time and finish time. The length refers to number of instruction per second.

2.2 Description of Algorithm

The tasks are sent to the broker, the broker determines the suitable VM based on a number of tasks that submitted as illustrated in Figure 1, and each VM state change among (Under loaded, Balance, High balance, Overloaded).

Scheduler: The user will send the Cloudlet_List, cloudlet independency list as input to the scheduler that enter the request to the balancing.

Balancing: It assigns the task to the suitable VM that is determined from that assigns tasks to VMs based on the information that is collected.

VMCheck: Check the VM state by comparing the capacity of VM with the VM load using three types of

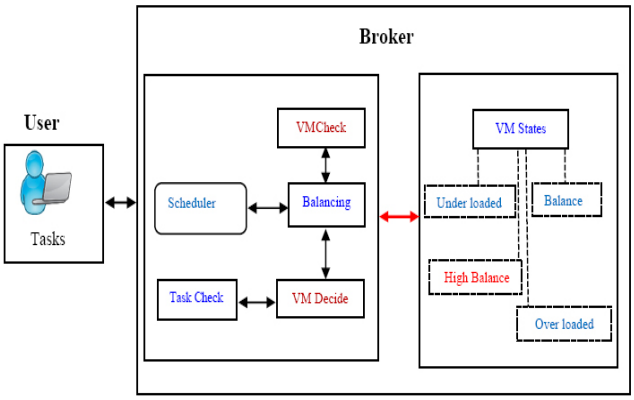

Figure 1. The proposed LBDA.

thresholds: threshold upper, threshold fair, and threshold lower.

The Capacity of VM (C_{VM}) is calculated based on Eq.1[1].

$$C_{VM} = Pe_{num} * Pe_{mips} \quad (1)$$

where, $Pe_{num}$is defined as a number of processing element allocated in VM.

Pe_{mips} is the amount of million instructions per second.

VM load (VM_{load}) that it will be calculated based on Eq.2[1].

$$VM_{loadi} = \frac{\sum_{j=1}^{n} TL_j}{n} \quad (2)$$

where, TL refers to task length.

Three thresholds were selected based on the previous process, first Threshold upper limited TUL, it is =0.9, the second Threshold for Fair Limited (TFL), it is =0.8 and the third Threshold for Lower Limited (TLL), it is =0.2, as illustrated in Figure 2 compared to VM load with a capacity of VM to update the VM state

Stage 1.$VM_{loadi} <= C_{VMi} * TLL$ *then VM state labelled as Under loadedstate.*

Stage 2. $VM_{loadi} > C_{VMi} * TLL$ *and* $VM_{loadi} <= C_{VMi} * TFL$ *then VM state labelled as Balanced state.*

Stage 3.$VM_{loadi} > C_{VMi} * TFL$ *and* $VM_{loadi} <= C_{VMi} * TUL$ *then VM state labelled as High Balancestate.*

Stage 4. $VM_{loadi} > C_{VMi} * TUL$ *then VM state labelled as Overloaded state.*

VM Decide: Selection of suitable VM will be decided and we take into account the benefit of user and provider, where increasing the resources utilization, load balancing and less completion time.

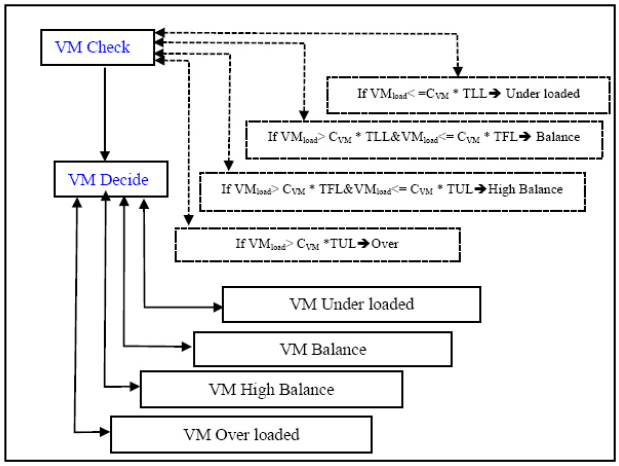

Figure 2. Virtual machine state.

First, we calculate the Estimated Completion Time (ECT) of the task in all virtual machines which have state (Under loaded), ECT calculated based on Eq.3[21] and Eq.4[22].

$$ET = \frac{TL}{C_{VM}} \quad (3)$$

$$\text{ECT} = ET + VM_{loadi} \quad (4)$$

ET is the execution time of task.

Then select VM that has less estimated completion time.

Second, if there are more than one VM Under state that has same estimated completion time then select VM with the largest capacity. Selection the largest capacity improves the load balancing because it decreases the load on VM that has less capacity and minimizing Over state happening.

Third, when all VMs' state is balanced and there are VMs becoming High Balance then Task Check must be taken into account to check the less estimated completion time of the task as follow:

a. We calculate the less ECT in all VMs' Balance state.
b. Less ECT in all VMs' High Balance state added with delay (ECT+ Delay).
c. If the less ECT in all VMs' state (Balance) more than the less ECT for VMs' state (High Balance) with delay then the task mapped to VM that has less ECT, but here the start execution time of task is equal to load of VM High Balance and the VM new state return to (Balance state).

Otherwise, send to the VM state (Balance) with less processing time and largest capacity, if there.

As mention in LBDA scheduling task algorithm avoidance the VM Overloaded state, so mapped the task with delay when the VM in a high balance state.

update VM load based on Eq.5.

$$VM_{loadi} = VM_{loadi} + TL \quad (5)$$

Finally, remove the task that is mapped from unmapped tasks list.

2.3 Algorithm Steps

Load Balancing Decision Algorithm (LBDA)

Input: List of un mapping tasks and available resources.
Output: Complete tasks with less execution time.

1 *Initially set the characteristics of VMs and create the tasks with different length.*

2 *Calculate VM capacity based on Eq. 1.*

For each task in tasks list:

1. Calculate VM load based on Eq. 2.

2. Check VM state:

2.1. If $VM_{Load} <= C_{VM} * TLL \rightarrow$ *means the state of the VM is*

Under loaded state.

2.2. If $VM_{Load} > C_{VM} * TLL \;\&\&\; VM_{Load} <= C_{VM} * TFL \rightarrow VM$

3 *means the state of the VM is Balance state.*

2.3. If $VM_{Load} > C_{VM} * TFL \;\&\&\; VM_{Load} <= C_{VM} * TUL \rightarrow$ *means the state of the VM is High Balance*

1. If $VM_{Load} > C_{VM} * TUL \rightarrow$ *means the state of the VM is Overloaded state.*

2. If there are VMs' Under loaded state, then calculates ECT based onEq.3 and mapping the task to VM that has less estimated completion time, if there are more than one VM has same ECT then selects VM with the largest capacity.

3. If all VMs' Balance then mapping task to VM has less estimated time and with largest capacity, if there.

4. If virtual machines are in the state (Balance and High Balance):

a. Calculate the less estimated completion time of VMs' Balance state, if there are more than one VM has same ECT then selects VM with the largest capacity.

b. Calculate the less estimated completion time of VMs' High Balance state with the delay.*

c. Finally, select the lowest of ECT between the two types VMs' state (Balance, High Balance)

** Delay:*

The delay means is the amount of time for completed tasks

process
when VM in High Balance loaded state
5. Update load ofVM based on Eq.5.
End for

4 *Calculate Average Makespan, Mean of Average Response Time and Total Execution Time.*

3. Performance Metrics

A set of parameters are considering when building task scheduling algorithm. These parameters play an important role to increase the overall cloud performance. Each parameter is explaining in this section.

Makespan: Is the total completion time of all tasks VM queue. A good scheduling algorithm always tries to reduce the makespan[23], which is defined in Eq.7[17].

$$Makespan = max\ \{CTi\} \tag{7}$$

where, i∈ VMs

then, calculating the average of makespan which is defined in Eq.8[24].

$$Av.\ Makespan = \frac{\sum_{k=1}^{m} Makespan}{m} \tag{8}$$

Response Time (RT): indicates to the time of search process, which contains the time to execute the task in cloud computing system[25]. Response Time (RT) is defined as Eq.9[16].

$$RT_j = FT_j - SB_j \tag{9}$$

where, FT_j is finish time of task, SB_j is submission time of taskthen calculate the Mean of Average Response Time of all VMs, defined as Eq.10.

$$M.\ Av.\ RT = \frac{\sum_{k=1}^{m} Av.RT_k}{m} \tag{10}$$

Av.RT is the average response time of VM denoted in Eq.11.

$$Av.\ RT = \frac{\sum_{j=1}^{n} RT_j}{n} \tag{11}$$

Execution Time: Is defined as the time difference between the task finish time and the task starting time within the resource. It is defined as Eq.12[25].

$$Execution\ Time_j = FT_j - ST_j \tag{12}$$

where, ST_j is start time of task.

4. Experimental Results

4.1 Simulation Tool

The simulator is used because all the new algorithms or strategies require being checked before applying them in the real cloud computing environment. But building a real world for cloud computing environment to check the newly proposed algorithms and policies wastes a lot of time and it gets too costly in terms of money as well. Thereby the cloud environment and run the policies are simulated on the simulator. The simulator is effective of the algorithms that can be detected with a very little cost of money and time[26].

We use a CloudSim 3.0.3, a CloudSim toolkit as a simulation platform that has been preferred, as it is a recent simulation framework embattled at cloud environments[27]. In CloudSim, the users are submitting the tasks or jobs. Classification of the user tasks are based on the parameters like memory, bandwidth, resources utilization and completion time etc. Then, the tasks are added to scheduling policies. After scheduling, CloudSim utilizes to assign the user tasks with virtual machine by datacenter broker and execute the tasks with the help of a virtual machine. CloudSim is implemented with supplying java[28].

4.2 Simulation Configuration

In the experiments, the simulation parameters are described as follows.

- Set the virtual machines with RAM 512 MB, size 10 GB, Bandwidth 1 GB and 300, 500 and 700 MIPS.
- Set the tasks Lengths with different length, file size 300 MB.
- Set the configurations for a host with RAM 16384 MB, Storage 1000 GB,

  Bandwidth 10 GB and 10000 MIPS.

Time-shared provisioning is used in CloudSim 3.0.3 toolkit for this implementation. The results for average makespan, Mean of Average Response Time and total all execution time that are shown in Table 1.

5. Performance Evaluation

Evaluated the results of the performance of the LBDA, Max-Min, SJF and Round Robin algorithms by using the CloudSim toolkit. The metrics have tested the Average of makespan (completion time of the last task

Table 1. All experiments result of LBDA for Av. Makespan, M.Av.RT and Total Execution Time

Experiments	Av.Makespan	M.Av.RT	Total Execution Time
100 Tasks in 6 VMs	360	277	29246
150 Tasks in 8 VMs	427	300	48335
200 Tasks in 10 VMs	430	313	71644
250 Tasks in 12 VMs	433	326	91165
300 Tasks in 14 VMs	458	332	112303

Table 2. Comparison of Av. Makespan among LBDA, Max-Min, SJF and RR

Experiments	Average Makespan			
	LBDA	Max-Min	SJF	Round Robin
100 Tasks in 6 VMs	358	370	418	416
150 Tasks in 8 VMs	416	438	488	492
200 Tasks in 10 VMs	430	460	524	501
250 Tasks in 12 VMs	434	460	522	521
300 Tasks in 14 VMs	458	488	549	550

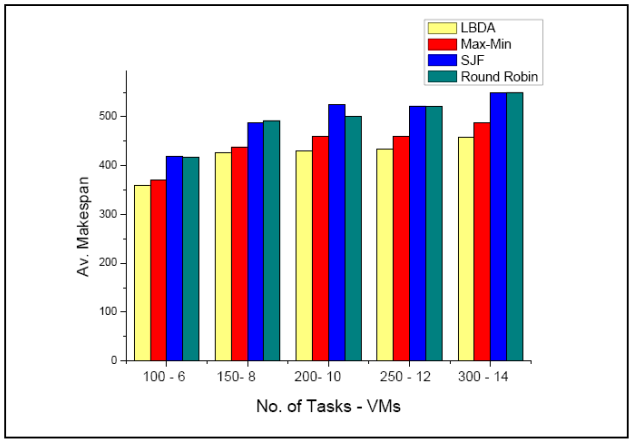

Figure 3. Average of Makespan

to be executed), the Mean of Average Response Time and Total of all Execution Time for all tasks in virtual machines. Five experiments 100, 150, 200,250 and 300 tasks executed in a 6, 8, 10, 12 and 14 number of VMs respectively.

Table 2, shows the results are implemented. It is obvious that average makespan based approach LBDA produced better results than the others algorithms in all experiments.

From Figure 3, it is visually obvious that minimizing average makespan in the proposed LBDA compared to others.

Table 3 and Figure 4 demonstrate the results of Mean of Average Response Time and the proposed LBDA had better results than Max-Min, SJF and Round Robin algorithms.

Also, Table 4 shown the proposed LBDA outperform results than Max-Min, SJF and Round Robin algorithms in Total Execution Time of all tasks. This mean that LBDA execute user tasks in less time when it used instead of existing algorithms.

From Figures 3, 4 and 5 present the Average Make span, the Mean of Average Response Time and Total Execution Time are decreased as we increase the number of tasks and number of virtual machines in our proposed algorithm LBDA and Max-Min, SJF, Round Robin algorithms.

Table 3. Comparison of Mean of Average Response Time among LBDA, Max-Min, SJF and RR

Experiments	Mean of Average Response Time			
	LBDA	Max-Min	SJF	Round Robin
100 Tasks in 6 VMs	277	300	340	349
150 Tasks in 8 VMs	308	356	397	414
200 Tasks in 10 VMs	320	374	426	500
250 Tasks in 12 VMs	336	375	424	436
300 Tasks in 14 VMs	338	462	447	396

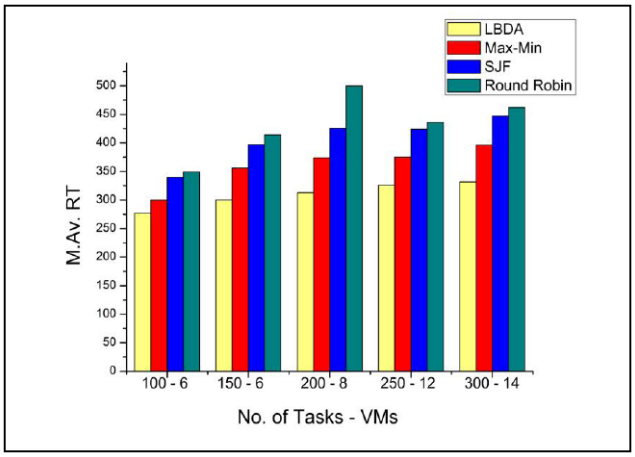

Figure 4. Mean of Average Response Time.

Table 4. Comparison of Total Execution Time among LBDA, Max-Min, SJF and RR

Experiments	Total Execution Time			
	LBDA	Max-Min	SJF	Round Robin
100 Tasks in 6 VMs	29456	30107	34224	35150
150 Tasks in 8 VMs	50683	53504	59609	62124
200 Tasks in 10 VMs	71676	75308	85304	100302
250 Tasks in 12 VMs	92008	94126	106418	109407
300 Tasks in 14 VMs	112956	119570	134135	138785

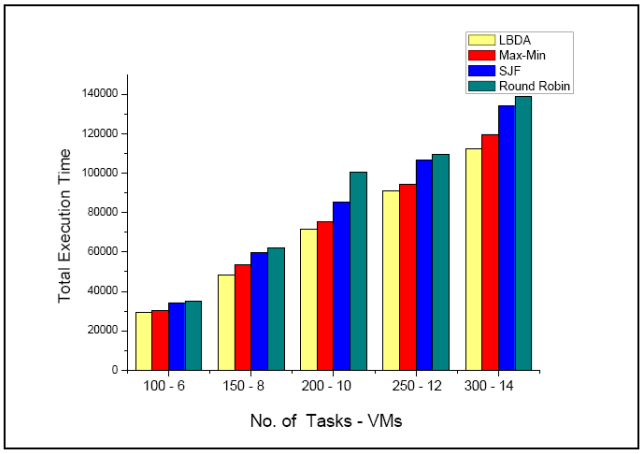

Figure 5. Total Execution Time.

All experiments illustrated that Load Balancing Decision Algorithm improves Average makespan, Mean of Average Response Time and Total Execution Time compared to other existing algorithms. In addition, increasing the number of VMs shows slight improvement. With this, we can deduce that LBDA gives better and efficient results.

6. Summary and Conclusion

In cloud computing environment, there are many challenges, one of them is load balancing. It has an important impact on the performance. The maximize resources utilization and users' satisfaction can be improved by good load balancing. In this proposed algorithm, LBDA solving the tasks in load balancing problem by considering the VMs' capacity and the estimated completion time for each task to map the tasks to the most appropriate VMs. We conducted five experiments to evaluate the performance of proposed algorithm. The comparison of the proposed algorithm is done with Round Robin, Max-Min and SJF algorithms under same configuration environment. The simulation results show that the proposed algorithm has outperformed in all cases as compared to Max-Min, SJF and Round Robin algorithms by reducing average makespan, Mean of Average Response Time and Total Execution Time in all VMs. As a part of the future, the implementation can be done for improvements with more factors such as a deadline constraint.

7. Acknowledgement

We thank Dr. Mokhtar Abdulrahman Alworafi and Dr. Asma Alhashmi for excellent comments and suggestions.

8. References

1. Chitra DD, Uthariaraj VR. Load balancing in cloud computing environment using Improved Weighted Round Robin Algorithm for nonpreemptive dependent tasks. The Scientific World Journal. 2016; 2016.
2. Solmaz A, Motamedi S, Sharifian S. Task scheduling using Modified PSO Algorithm in cloud computing environment. International Conference on Machine Learning, Electrical and Mechanical Engineering; 2014. p. 37–41.
3. Imran MA, Pandey M, Rautaray SS. A proposal of resource allocation management for cloud computing. International Journal of Cloud Computing and Services Science. 2014; 3(2):79–86.
4. Jamuna RMR, Gouda KC, Nirmala N. Load balancing technique for climate data analysis in cloud computing environment. International Journal of Science, Engineering and Computer Technology. 2013; 3(5):183–85.
5. Namrata G, Garala K, Maheta P. Cloud load balancing based on ant colony optimization algorithm. IOSR Journal of Computer Engineering (IOSR-JCE); 2015. p. 11–18.
6. Danilo A. Quality-of-service in cloud computing: modeling techniques and their applications. Journal of Internet Services and Applications. 2014; 5(1):1–17.
7. Jia Z. A Heuristic clustering-based task deployment approach for load balancing using Bayes Theorem in cloud environment. IEEE Transactions on Parallel and Distributed Systems. 2016; 27(2):305–16. Crossref
8. Kunjal G, Goswami N, Maheta ND. A performance analysis of load Balancing algorithms in Cloud environment. 2015 International Conference on Computer Communication and Informatics (ICCCI), IEEE; 2015. p. 4–9.
9. Beghdad BK, Benhammadi F, Benaissa F. Balancing heuristic for independent task scheduling in cloud computing. 2015 12th International Symposium on Programming and Systems (ISPS), IEEE; 2015.
10. Aditi S, Sharma S. Credit based scheduling using deadline in cloud computing environment. International Conference on Resent Innovation in Science Engineering and Management; 2016. p. 208–16.
11. Sukhjinder GS, Vivek T. Implementation of a hybrid load balancing algorithm for cloud computing. International Conference on Science, Technology and Management; 2016. p. 173–82.
12. Mohana PS, Subramani B. A new approach for load balancing in cloud computing. International Journal of Engineering and Computer Science. 2013.
13. Shreya S, Kaur A. Load balancing in cloud computing using Shortest Job First and Round Robin Approach. International Journal of Science and Research. 2015; 9(4):1577–80.
14. Divya C, Chhillar RS. A new load balancing technique for virtual machine cloud computing environment.

International Journal of Computer Applications. 2013; 69(23):37–40. Crossref
15. Yang X, HongTao L. Load balancing of virtual machines in cloud computing environment using improved ant colony algorithm. International Journal of Grid and Distributed Computing. 2015; 8(6):19–30. Crossref
16. Abbas RH, Katti CP, Saxena CP. A load balancing strategy for Cloud Computing environment. 2014 International Conference on Signal Propagation and Computer Technology (ICSPCT), IEEE; 2014.
17. Babu DL, Venkata PK. Honey bee behavior inspired load balancing of tasks in cloud computing environments. Applied Soft Computing. 2013; 13(5):2292–303. Crossref
18. Elhossiny I, El-Bahnasawy N, Omara FA. Job scheduling based on harmonization between the requested and available processing power in the cloud computing environment. International Journal of Computer Applications. 2015; 125(13):1–4.
19. Elrasheed I, Alamri F. Optimized load balancing based task scheduling in cloud environment. International Journal of Computer Applications; 2014.p. 35–8.
20. Ali A, Omara FA. Task scheduling using hybrid algorithm in cloud computing environments. IOSR Journal of Computer Engineering. 2015; 17(3):96–106.
21. Sourav B. Development and analysis of a new cloudlet allocation strategy for QoS improvement in cloud. Arabian Journal for Science and Engineering. 2015; 40(5):1409–25.Crossref
22. Nizomiddin BK, Choe TY. Dynamic task scheduling algorithm based on ant colony scheme. International Journal of Engineering and Technology (IJET). 2015; 7(4):1163–72.
23. Saleh A, Yussof S, Ezanee M, Almiani M. A review energy-efficient task scheduling algorithms in cloud computing. Long Island Systems, Applications and Technology Conference (LISAT); 2016.
24. Vinay D, Shah J, Mehta R. Dynamic load balancing for cloud computing using heuristic data and load on server. IOSR Journal of Computer Engineering (IOSR-JCE). 2014; 16(4):59–69. Crossref
25. Hussain MSH, Latiff MSA, Coulibaly Y. An appraisal of meta-heuristic resource allocation techniques for IaaS cloud. Indian Journal of Science and Technology. 2016; 9(4):1–14.
26 Kritika S, Maini R. Comparative analysis of host utilization thresholds in cloud datacenters. International Journal of Computer Applications. 2015; 120(2):9–13. Crossref
27. Lodhi V, Sarveshrai, Vishwakarma GK, Enhanced minimum utilization VM selection mechanism for clouds. International Journal of Computer Science and Information Technologies. 2015; 6(3):2975–77.
28. Dinesh K, Poornima G, Kiruthika K. Efficient resources allocation for different jobs in cloud. International Journal of Computer Applications. 2012; 56(10):30–5. Crossref