Pertanika J. Sci. & Technol. 28 (3): 1049 - 1061 (2020)

PERTANIKA JOURNALS

SCIENCE & TECHNOLOGY

Journal homepage: http://www.pertanika.upm.edu.my/

Load Balancing using Enhanced Multi-Objective with Bee Colony Optimization in Cloud Networks

Abhikriti Narwal* and Sunita Dhingra

Department of Computer Science and Engineering, University Institute of Engineering and Technology, Maharshi Dayanand University, Rohtak Haryana, India

ABSTRACT

In the course of recent decades, cloud computing has turned into a hot research point for the logical, scholastic and mechanical networks. It is a wide-ranging term used to depict an extra class of framework-based enrolling that occurs over the web. The distributed computing principally plans to give capable access to remote and geologically disseminated assets. The other important purpose of cloud service providers is to gain maximum profit and use resources efficiently. As cloud technology is evolving day by day and confronts numerous challenges, one of them being uncovered is scheduling. Scheduling refers to a set of policies to control the order of work to be performed by a system. Every task needs a scheduling strategy which is assigned by the system in order to get executed by the processor. Procedures are vigorous to plan the trades for accomplishment. Job scheduling procedures supposed to be the most assumed difficulties in the cloud computing domain. The survey of existing papers reveals the better makespan time but cannot guarantee the proper balancing of load. To overcome this issue, Enhanced Multi-Objective Load balancing Scheduling Algorithm (EMOLB_LB) is proposed which uses Bee Colony Optimization algorithm for the analysis and balancing of load with more objective functions to sort the tasks and improvise the performance in terms of cost and time. The existing scheduling technique, Enhanced Multi-Objective Scheduling Algorithm (EMOSA) uses only non-dominating strategy for sorting the tasks but load management is not taken into consideration which is further optimized by proposing EMOLB_LB

ARTICLE INFO

Article history:
Received: 28 November 2019
Accepted: 12 March 2020
Published: 16 July 2020

E-mail addresses:
abhikritiin@gmail.com (Abhikriti Narwal)
sunitadhingramdu@rediffmail.com (Sunita Dhingra)
*Corresponding author

ISSN: 0128-7680
e-ISSN: 2231-8526

technique. The experimental results were analysed and compared with various existing techniques like Multi Objective Scheduling algorithm (MOSA), EMOSA and showed that the EMOLBA_LB technique was better than the earlier techniques in term of each performance attribute like average waiting time by 2.934 %, processing cost by 17.6% and processing time by 20.5%.

Keywords: Bee colony optimizations, cloud computing, MOSA (Multi Objective Scheduling algorithm)

INTRODUCTION

Cloud computing is an extension of parallel computing, distributed computing and grid computing[1]. The most recent movements in distributed framework are able to assemble our expert to offer a progressively versatile system. Distributing processing is the premature advancement which depends upon pay per-use criteria. It is enlisting point of view where requests, information, data transmission and IT associations are given over the Internet. The objective of the cloud association suppliers is to utilize assets effectively and accomplish the most phenomenal favourable position. The enduring evolution of cloud computing in IT has led several explanatory remarks on cloud computing. The US National Institute of Standards and Technology (NIST) defines the cloud computing as:"Cloud computing is a model enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction" (Narwal & Dhingra, 2016). There are hundred millions of gadgets associated with the web that are using a considerable amount of distributed computing administrations day by day as it provides a flexible and easy way to keep and retrieve data and files. Distributed computing is a promising and up-coming advancement which allows the customers to pay as they require. It engages encouraging of inescapable applications from client, exploratory, and business regions. Distributed computing is advancing utility-arranged IT organizations to customers around the globe (Foster et al., 2008). The creating cost of tuning and managing PC structures is provoking out-sourcing of business organizations to core interests. The features of distributed framework fuse on self-association, broad structure, asset pooling, smart flexibility and assessed association (Xiao et al., 2014). On intrigue self-association, it recommends that clients (ordinarily affiliations) can ask for and deal with their own particular computing assets. Distributed framework is an accumulation of two phrasings in the situation of figuring innovation with computing resources. It is an investor of diverse assets.

Scheduling refers to the way of assigning errands to resources that require the quality constraints (Singh et al., 2014). As countless customers offer cloud assets and dispatch

1 http://www.cncloudcomputing.co.

their assignments to the cloud, it has turned into a test to plan these undertakings. Thus, work planning is a hotly debated issue in conveyed and distributed computing. It is the procedure through which the processes and threads are mapped to resources by utilizing the accesses (Tripathy & Patra, 2014). The necessity of scheduling appears due to the prerequisite of performing multitasking and multiplexing. If scheduling is done properly then they can boost the interpretation of a system (Bhatia & Sharma, 2015). It is singular significant factor for providing better performance by a system. Scheduling refers to the planning or relegating a job to Virtual Machines (VM) in such a way keeps the usage upsurge (Narwal & Dhingra, 2016). A skillful scheduling scheming progresses the wide-ranging simulator presentation and supports facility of supplier to furnish fine quality parameters constraint. An elite VM dole out with the high-quality constraint. Data Centre Broker takes the solicitations from the client and transfers the solicitation to the VM which best suites the need and quality of service contention. Piecemeal a decent quality task is allotted to a low Quality of Service (QoS) VM which prompts the unfortunate procedure of assets and this abuses the Service level contention (Kowsik & Rajakumari,2014). So, an effective task scheduling calculation ought to be executed at the broker.

RELATED WORK

There are different studies adjacent the utilization of headway techniques of scheduling in cloud computing. To resolve the issues of existing procedures and falling the completion, an optimized priority based method has been proposed by Ghanbari and Othman (2012), taking into consideration of various principles and decision making rules to find out the job which is to be mapped to the specific VM by using various QoS parameters. In 2012, Zhan and Huo had projected an assessment relating to the utilization of improvised Particle Swarm

Optimization (PSO) solidified with Scheduling Algorithm (SA) in scheduling the resources of distributed systems to reorganize the scheduling procedure, by evolving the joining pace and using extent of compensations. Lakra and Yadav (2015) had introduced a scheduling procedure using multi objective functions to check the task dominance with purpose of mapping the task to the machine which would enhance throughput of the scheduler by reducing the completion time. To demonstrate the competence of the procedure based on optimization, Mathukiya and Gohel (2015) had discussed the multi-objective task scheduling algorithm for improvising the output of the scheduler and introduced non-dominated sorting for requesting of tasks. In order to minimize the expense of the processing, Guo et al. (2012) had defined a model for user's requirements scheduler in cloud framework taking into account on heuristic calculation based on the behaviour of particles, rely on small portion value rule. Patel and Bhoi (2014) had projected a precedence based scheduling procedure keeping with the purpose of accomplish better makespan and consistency of jobs by utilizing iterative system. Multi criteria and numerous

attribute decision making model were utilized to achieve better execution. Singh and Kalra (2014) had introduced an intricate thought regarding Genetic Algorithm (GA) and its various versions propounded as scheduling of user's requirement in cloud framework and changed the initial population of GA using Enhanced Max-Min which reduces the finish time of all the jobs and balanced the load as well. Lawrence and Silas (2013) proposed a resource scheduling methodology using potentially all pair-wise rankings of all possible alternatives (PAPRIKA). PAPRIKA method evaluates fairness based on user satisfaction. This method makes use of both task matrix and resource matrix. Priority of the resource is calculated with respect to a threshold value. Tasks are then mapped to the resource that gives higher user satisfaction, thus improving resource utility and minimizing the allocation time. Abdullah and Othman (2013) tried to use the Divisible Load Theory (DLT) to design efficient strategies to minimize the overall processing time for scheduling jobs in cloud computing. In this research homogeneous processors are considered and a closed form solution is derived to assign fractions to each processors. The research has tried to schedule the jobs in such a way that quality of service (QOS) parameters can achieve maximum benefit and results in minimizing the overall total cost. Bini and Sindhu (2015) propounded Hyper-Heuristic scheduler on Cloud constructed assemblies. Optimization procedures of GA and SA are in the solution of procedures pool and can be considered as traditional heuristic procedures. In addition, the combined solution of Differential Algorithm (DA) and GA minimizes the execution and the makespan (Bini & Sindhu, 2015). Kanani and Maniyar (2015) had inspected a transitory survey of Max-Min scheduler. The paper had propounded procedure which was familiar to evade downsides of Max-Min calculation to decrease the finish time and manufactured the asset utilization with seeing customer need, so that the prioritize task was accomplished foremost as demonstrated by its need. Job with better priority would be accomplished foremost than other subordinate exertion needs with the target that user's superlative position could be achieved all the additional total (Kanani & Maniyar, 2015). Raja and Sekar (2016) ran a better credit-based scheduling algorithm by means of the constraints like user priority, task length and deadline constraints. The new results show a significant improvement in the consumption of resources with speedy response time. Within the experimental result, the projected rule shows higher result than the present rule. Credits area unit accustomed cut back the build span of the task and execute all the task in cloud (Ru & Keung, 2013). Zalavadiya and Vaghela (2016) had projected technique showing the priority tasks area unit far from the overladen virtual machine and that they area unit allotted to below loaded virtual machine. It helps to scale back the minimum completion time, quantity of waiting time of tasks in queue is token and win higher resource utilization. This paper concludes that the minimum quantity of your time is taken to execute the tasks and higher resource utilization (Silberschatz et al, 2014). Shameer and Subhajini (2017) had incontestable the correct use of load reconciliation

techniques to extend the performance of the system and cut back the price and energy. Increased Bee Colony rule primarily based multi-objective task planning technique is given and its performance is verified and tested by the CloudSim stimulator. Also, through the experimental analysis, the simplest performance is shown for the projected work (Raja & Sekar, 2016). Vijay and Ghita (2017) had mentioned about the operating of planning algorithms which discusses about the shortest job initial, timeserving load reconciliation and generalised priority rule. They were placed to check the various condition and things and were assessed addicted to parameters, as an example, cost and makespan (Zalavadiya & Vaghela, 2016).The author has explained improvised multi objective scheduling by assigning the job using non dominating sort to the best optimized machine in the list to achieve the best results by diminishing processing time of the solution (Narwal & Dhingra, 2019). Narwal and Dhingra (2017) had discussed the comparative analysis of EMOSA and credit based scheduling algorithm. These algorithms are analysed based on performance parameters and on different scenarios. Also explained that optimization algorithms can be combined to make the results more accurate (Vijay & Ghita, 2017).

METHODOLOGY

The concept has been taken from Shameer and Subhajini (2017). EMOSA gained the best results of scheduling by comparing it with other scheduling procedures, but still it does not take into consideration of load balancing while mapping the machines to the tasks. EMOSA algorithm works with the principle of non-dominating sorting i.e. the tasks are sorted on the basis of multiple objective functions including length, file size and output file size of the task. It uses these objectives to check which task dominates the other tasks and based on this criterion, the best one is ranked highest among all and all other tasks are sorted in this order (Narwal & Dhingra, 2017). Then the organized errands are planned to the virtual machine. The arrangement of these enormous number of undertakings is an amazingly essential and challenging exertion for cloud. The crucial point of planning is to achieve cloud accomplishment of jobs to the extent of recovering output, load adjustment, QoS, financial practicality and the ideal activity time. The concept of load balancing was not taken into consideration i.e. whether that particular virtual machine to the cloudlet is going to be mapped is overloaded or underloaded. Therefore, in order to achieve efficient consumption of cloud resources, the problem of load balancing needed to be resolved. For this, the concept of load balancing was merged with the scheduling algorithm by introducing a Honey Bee Optimization algorithm to achieve the balancing of virtual machines.

In this paper, before mapping the tasks to the virtual machine, load on that particular machine was calculated. Based on the load, underloaded and overloaded machines were evaluated then their demands and supplies were calculated which helped to assign the various tasks to the machine. For the balancing of load, Bee Colony optimization approach was used; in which bees were tasks and the food for the bees was the virtual machines i.e. need to map bees with their best food nest.

Proposed Algorithm

Load Balancing using Bee colony approach works in four different modules:
1. Calculating Load of VM

$$\text{Load}_i = \frac{\text{N} * \text{Length of tasks}}{\text{VM\_MIPS}}$$

Here N is the Number of jobs and VM_MIPS is the Million Instructions per second rate of the virtual machines.

a)VM Capacity is evaluated using:

$$Capacity = No\ of\ PE_s * MIPS\ of\ PE_s + Bandwidth\ of\ VM_s$$

b) Load and Capacity of Virtual Machine can be premeditated from the given two equations:

$$\text{LOAD}_{\text{datacenter}} = \sum \text{Load}$$
$$\text{CAPACITY}_{\text{datacenter}} = \sum \text{Capacity}$$

c)Using the equations below processing time of virtual machine and datacenter (PTdatecenter) can be premeditated:

$$\text{PT}_{\text{vm}} = \frac{\text{Load}}{\text{Capacity}}$$

$$\text{PT}_{\text{vm}} = \frac{\text{Load}_{\text{datacenter}}}{\text{Capacity}_{\text{datacenter}}}$$

d) Standard Deviation (SD) for the load can be evaluated using

$$SD = \sqrt{\frac{1}{N}\sum_{i=0}^{N}(X_i - \overline{X})^2}$$

Where X_i is Processing Time and $\overline{X}$ is the average Processing Time of the virtual machine

2) Using the above SD calculations, comparison of SD with some threshold value rabges between 0-1, can define whether the machine needs load balancing or not. Following conditions defines the procedure for the same.

if (SD>threshold) {
Overloaded =True; load balancing is needed as the machine is overloaded.}
Else if (SD<threshold) {
Underloaded =True; load balancing is not needed here.}

3) Overloaded machines search for the underloaded machines to share their load i.e. they demand for the capacity for machine which can take the overloaded tasks

Demand_OverLoadedVM = Load – Capacity_VM

4) Underloaded machines search for the overloaded machines to take the load of overloaded machine i.e. they supply the capacity which can be taken by the overloaded tasks

Supply_OverLoadedVM = Capacity_VM-Load

RESULT AND DISCUSSIONS

The proposed approach had been simulated using JAVA JDK Netbeans IDE with cloudsim 3.0 simulator. The results had been analysed on various workloads with different number of virtual machines and cloudlets compared with other existing techniques including First Come First Serve (FCFS), Shortest-Job-First (SJF), MOSA (multi-objective scheduling algorithm), EMOSA (Enhanced multi-objective scheduling algorithm) in terms of average waiting time, total processing cost and total processing time (Table 1).

Table 1

List of Constraints for Investigation of Consequences

CloudSim Objects	Input Specifications	Value
Job	Len_Cloudlet	100-20000
	Number._of_Cloudlets	10-2000
Virtual Machine	Count_of_Vm	3-100
	MIPS_Vm	250-2000
	VM_Memory	512-2048
	VM_BW	500-1000
	Pes_Count	1-4
Server (Datacenter)	Number_of._Datacenters	2-5
	Hosts_Count	2-6

Analysis in terms of Average Waiting Time

Average Waiting Time (AWT) is the time the task has to wait for the virtual machine. Average waiting Time is defined using the below Equation 1:

$$\frac{\sum_{i=0}^{no\ of\ cloudlets} cloudlet.waitingtime()cloudlet.WaitingTime}{No.of\ Cloudlets} \quad (1)$$

Figure 1 and Table 2 show the comparison results of Average Waiting Time of the propounded algorithm EMOLB_LB with other existing algorithms. Figure 1 shows that the proposed algorithm performs better than the other procedures. For workload with 60 vms and 500 cloudlets, the AWT is 0. 4913 ms , EMOSA (Narwal & Dhingra, 2017) is having

Table 2

Simulation results of EMOSA_LB in terms of AWT

[Virtual Machine, Cloudlets]	First Come First Serve	Shortest Job First	Multi Objective Scheduling	E-Multi Objective Scheduling	EMOLB_LB
[3,9]	0.7591	0.6598	0.4469	0.4067	0.3625
[3,50]	0.7096	0.5964	0.4284	0.3947	0.3796
[30, 200]	1.98694	0.8026	0.5694	0.4954	0.4498
[40, 300]	1.42684	0.8569	0.5967	0.5036	0.4874
[60, 500]	1.52146	0.8832	0.5712	0.5169	0.4913

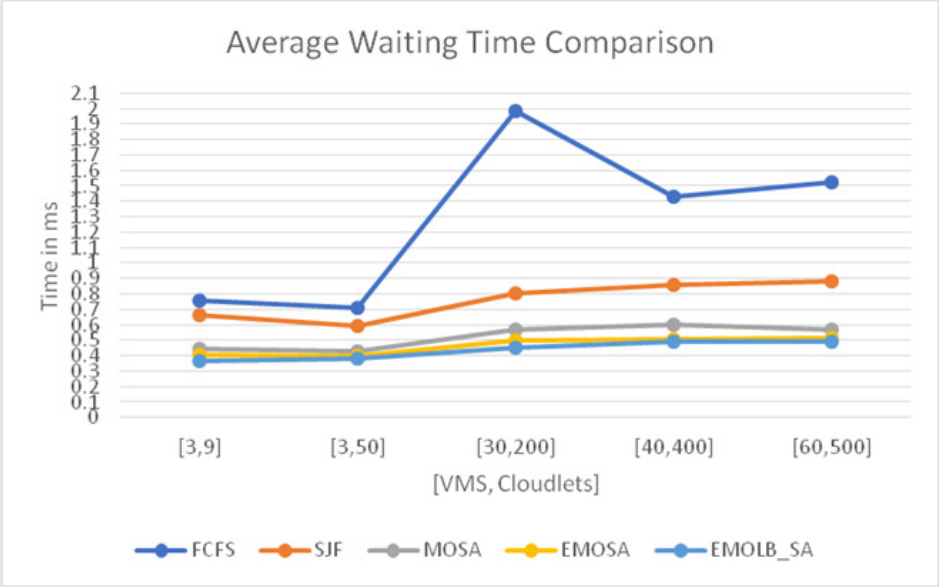

Figure 1. AWT of EMOLB_SA in contrast with EMOSA, MOSA, FCFS and SJF

0.5169 ms, MOSA (Lakra & Yadav, 2015) is 0.5712 ms, SJF (Ru & Keung, 2013) is 0.8832 ms and FCFS (Silberschatz et al., 2014) is 1.52146 ms. Similarly, for other workloads, the propounded EMOLB_LB performs better than other methods. The average waiting time improvement over the existing EMOSA scheduling technique is 2.934% approximately.

Analysis in terms of Processing Cost

It is the rate essential by the task to accomplish on the machine. Total Processing Cost can be denoted by below Equation 2

$$Total\ Processingcost = \sum DataCenterCharateristics_{costpermemory} * VM_{RAM} \quad (2)$$

Table 3

Simulation consequences of EMOSA_LB in relation to processing cost

[Virtual Machine, Cloudlets]	First Come First Serve	Shortest Job First	Multi Objective Scheduling	E-Multi Objective Scheduling	EMOLB_LB
[3,9]	309.9	298.3	220.18	117.9	107.7
[3,50]	596.36	546.36	524.6	315.3	211.7
[30, 200]	8569.3	6895.3	6369.365	6132.69	5880.94
[40, 300]	14012.05	10315.05	9704.36	9584.78	7232.99
[60, 500]	38569.75	21600.56	18695.34	18201.69	17037.92

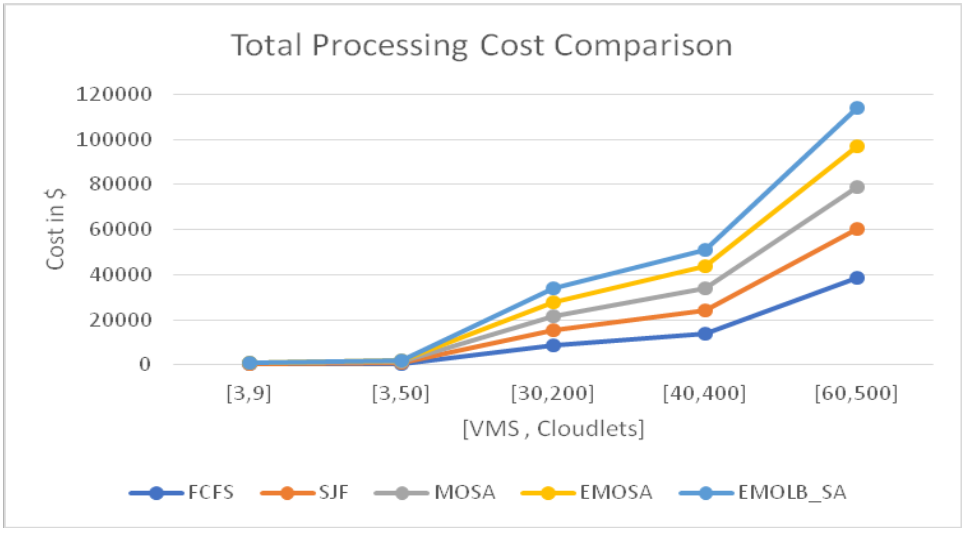

Figure 2. Total processing cost of EMOLB_SA with EMOSA, MOSA SJF and FCFS

Figure 2 and Table 3 show the comparison results of Total Processing Cost of the propounded algorithm EMOLB_LB with other existing algorithms. Figure 2 shows that the proposed algorithm performs better than the other procedures. For workload with

60 vms and 500 cloudlets, the TPC is 17037.92, EMOSA (Narwal & Dhingra, 2017) is having 18201.69, MOSA (Lakra & Yadav, 2015) is 18695.34, SJF (Ru & Keung, 2013) is 21600.56 and FCFS (Silberschatz et al., 2014) is 38569.75. Similarly, for other workloads, the propounded EMOLB_SA performs better than other methods. The Total Processing Cost improvement over the existing EMOSA scheduling technique is 17.6% approximately.

Analysis in terms of Processing Time

It is the period occupied by the machine to accomplish the job on particular virtual machine. Total Processing time is depicted using the below mentioned Equation 3

$$Total\ Processing_{time} = \sum cloudlet_{length} / (vm_{MIPS} * no_{ofPES}) \quad (3)$$

Table 4

Simulation consequences of EMOSA_LB in relation to processing time

[Virtual Machine, Cloudlets]	First Come First Serve	Shortest Job First	Multi Objective Scheduling	E-Multi Objective Scheduling	EMOLB_LB
[3,9]	4003.64	3994.37	3745.89	3438.65	1402.055
[3,50]	24778.39	23965.54	22648.31	21325.32	10792.71
[30, 200]	196547.2	189624.2	151847.73	124920.7	101918.2
[40, 300]	532652.37	465287.61	386547.97	346215.4	131059.3
[60, 500]	958634.62	935697.25	9.16E+05	894631.17	142625.8

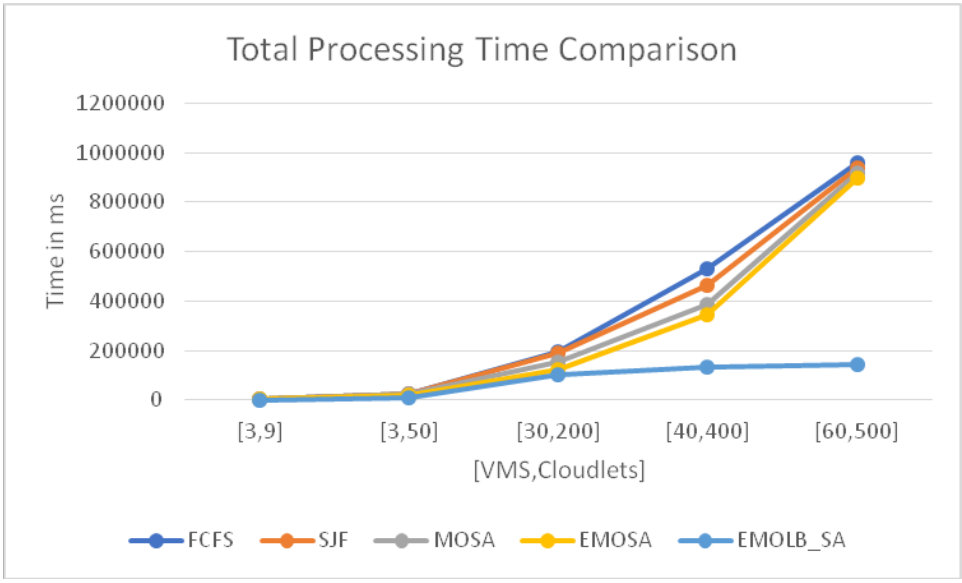

Figure 3. Total processing time of EMOLB_SA with EMOSA, MOSA SJF and FCFS

Figure 3 and Table 4 show the comparison results of Total Processing Time of the propounded algorithm EMOLB_LB with other existing algorithms. Figure 3 shows that the propounded algorithm performs better than the other procedures. For workload with 60 vms and 500 cloudlets, the TPT is 142625.8 ms, EMOSA (Narwal & Dhingra, 2017) is having 894631.17 ms, MOSA (Lakra & Yadav, 2015) is 916358.66 ms, SJF (Ru & Keung, 2013) is 935697.56 ms and FCFS (Silberschatz et al., 2014) is 958634.62 ms. Similarly, for other workloads, the propounded EMOLB_LB performs better than other methods. The Total Processing time improvement over the existing EMOSA scheduling technique is 20.5% approximately.

CONCLUSION

In this paper an effective adjustment of Enhanced Multi-objective task algorithm with Load balancing for a task scheduling has been done. The propounded algorithm is replicated in cloudsim simulator using netbeans IDE and shows that the results are better than the existing procedures Various calculations had been performed in order to minimize the aggregate processing time, cost and average waiting time of a given arrangement of tasks and comparisons of results had been performed with already existing algorithms. The Proposed algorithm increases the efficiency, throughput and accuracy of resource utilization on the cloud network. This proposed technique is ideal for task scheduling which provides the best scheduling solution in an optimal processing time. The future extension is to comprehend and to enhance the proposed calculation by using resource aware and more load balancing algorithms.

ACKNOWLEDGEMENT

The authors are sincerely thankful to the Vice Chancellor and Registrar of Maharshi Dayanand University, Rohtak for all the support to pursue the Doctorate of Research.

REFERENCES

Abdullah, M., & Othman, M. (2013). Cost-based multi-QoS job scheduling using divisible load theory in cloud computing. *Procedia Computer Science, 18*, 928-935.

Bhatia, A., & Sharma, R. (2015). An analysis report of workflow scheduling algorithm for cloud environment. *International Journal of Computer Applications, 119*(12), 21-25.

Bini, B. T., & Sindhu, S. (2015). Scheduling in cloud based on hyper-heuristics. International *Journal for Research in Applied Science & Engineering Technology, 3*, 380-383.

Foster, I., Zhao, Y., Raicu, I., & Lu, S. (2008, November 12-16). Cloud computing and grid computing 360-degree compared. In *2008 Grid Computing Environments Workshop* (pp. 1-10). Austin, TX, USA.

Ghanbari, S., & Othman, M. (2012). A priority based job scheduling algorithm in cloud computing. *Procedia Engineering, 50*(0), 778-785.

Guo, L., Zhao, S., Shen, S., & Jiang, C. (2012). Task scheduling optimization in cloud computing based on heuristic algorithm. *Journal of Networks, 7*(3), 547-553.

Kanani, B., & Maniyar, B. (2015). Review on max-min task scheduling algorithm for cloud computing. *Journal of Emerging Technologies and Innovative Research, 2*(3), 781-784.

Kowsik, P., & Rajakumari, K. (2014). A Comparative Study on Various Scheduling Algorithms in Cloud Environment. *International Journal of Innovative Research in Computer and Communication Engineering, 2*(11), 112-121.

Lakra, A. V., & Yadav, D. K. (2015). Multi-objective tasks scheduling algorithm for cloud computing throughput optimization. *Procedia Computer Science, 48*, 107-113.

Lawrance, H., & Silas, S. (2013). Efficient Qos based resource scheduling using PAPRIKA method for cloud computing. *International Journal of Engineering Science and Technology, 5*(3), 638-643.

Mathukiya, E. S., & Gohel, P. V. (2015). Efficient QoS Based Tasks Scheduling using multi-objective optimization for cloud computing. *International Journal of Innovative Research in Computer and Communication Engineering, 3*(8), 7169-7173.

Narwal, A., & Dhingra, S. (2016). A systematic review of scheduling in cloud computing framework. *International Journal of Advanced Studies in Computers, Science and Engineering, 5***(7)**, 1-9.

Narwal, A., & Dhingra, S. (2017). Enhanced task scheduling algorithm using multi-objective function for cloud computing framework. In *International Conference on Next Generation Computing Technologies* (pp. 110-121). Singapore: Springer.

Narwal, A., & Dhingra, S. (2019). Performance analysis of multiobjective algorithms for cloud computing framework. *Journal of Advanced Research in Dynamical and Control Systems, 11*(05-Special Issue), 2093-2099.

Patel, S. J., & Bhoi, U. R. (2014, August 27-29). Improved priority based job scheduling algorithm in cloud computing using iterative method. *In 2014 Fourth International Conference on Advances in Computing and Communications* (pp. 199-202). Cochin, India.

Raja, K., & Sekar, G. (2016). An algorithm for credit based scheduling in cloud computing environment depending upon deadline strategy. *IOSR Journal of Computer Engineering (IOSR-JCE), 2*(8), 70-76.

Ru, J., & Keung, J. (2013, June 4-7). An empirical investigation on the simulation of priority and shortest-job-first scheduling for cloud-based software systems. In *2013 22nd Australian Software Engineering Conference* (pp. 78-87). Melbourne, VIC, Australia.

Shameer, A. P., & Subhajini, A. C. (2017, September 8-9). Throughput maximization on efficient load balancing in cloud task scheduling using enhanced bee colony algorithm. In *2017 International Conference on Current Trends in Computer, Electrical, Electronics and Communication (CTCEEC)* (pp. 294-298). Mysore, India.

Silberschatz, A., Galvin, P. B., & Gagne, G. (2014). *Operating system concepts essentials*. Hoboken, New Jersey: John Wiley & Sons, Inc.

Singh, R. M., Paul, S., & Kumar, A. (2014). Task scheduling in cloud computing. *International Journal of Computer Science and Information Technologies (IJCSIT), 5*(6), 7940-7944.

Singh, S., & Kalra, M. (2014). Task scheduling optimization of independent tasks in cloud computing using enhanced genetic algorithm. *International Journal of Application or Innovation in Engineering and Management, 3*(7), 2319-4847.

Tripathy, L., & Patra, R. R. (2014). Scheduling in cloud computing. *International Journal on Cloud Computing: Services and Architecture (IJCCSA), 4*(5), 21-7.

Vijay, Y., & Ghita, B. V. (2017, July). Evaluating cloud computing scheduling algorithms under different environment and scenarios. In *2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT)* (pp. 1-5). IEEE.

Xiao, P., Hu, Z., Liu, D., Zhang, X., & Qu, X. (2014). Energy-efficiency enhanced virtual machine scheduling policy for mixed workloads in cloud environments. *Computers and Electrical Engineering, 40*(5), 1650-1665.

Zalavadiya, K., & Vaghela, D. (2016). Honey bee behavior load balancing of tasks in cloud computing. *International Journal of Computer Applications, 139*(1), 16-19.

Zhan, S., & Huo, H. (2012). Improved PSO-based task scheduling algorithm in cloud computing. *Journal of Information and Computational Science, 9*(13), 3821-3829.