ISSN 2320-9984
Volume 6 Issue 4 | December 2018 ISSN: 2320-9984 (Online)
International Journal of Modern Engineering & Management Research
Website: www.ijmemr.org

Enhanced Task Scheduling Algorithm of Cloud Computing Environment and Evaluation of Resource Provisioning Algorithm

Shriram Dhurwey
Assistant Professor
Department Information Technology
Jabalpur Engineering College
Jabalpur (M.P.), [India]
Email:srdhurve@jecjabalpur.ac.in

Neelu Tiwari
Assistant Professor
Department of Computer Science & Engineering
Hitkarini College of Engineering and Technology
Jabalpur (M.P.), [India]
Email:dubeyneelu31@gmail.com

Abstract**—***The extensive experiments demonstrate that the proposed Max-Min task-scheduling algorithm can improve the resource utilization as well as reduce the response time of tasks. A Max-Min task-scheduling algorithm for load balance in the elastic cloud is proposed in this paper. To realize the load balancing, the proposed algorithm maintains a task status table to estimate the real-time load of virtual machines and the expected completion time of tasks, which can allocate the workload among nodes and realize the load balance. Cloud Computing is the use of computing resources (Hardware and Software) that are delivered as a service over a network (typically the internet). It supplies a high performance computing based on protocols which allow shared computation and storage over long distances. In cloud computing, there are many tasks requires to be executed by the available resources to achieve best performance, minimal total time for completion, shortest response time, utilization of resources etc. Because of these different intentions, we need to design, develop, propose a scheduling algorithm to outperform appropriate allocation map of tasks on resources. A unique modification of Improved task scheduling algorithm is proposed. The algorithm is built based on comprehensive study of the impact of Improved task scheduling algorithm in cloud computing.*

Keywords:— *Cloud Computing, Job Scheduling, Makespan, Load balancing, Minimum completion time, Minimum execution time, Service response time.*

1. INTRODUCTION

Cloud Computing is getting advanced day by day. Cloud service providers are willing to provide services using large scale cloud environment with cost effectiveness. Also, there are some popular large scaled applications like social-networking and e-commerce. These applications can benefit to minimize the costs using cloud computing. Cloud computing is considered as internet based computing service provided by various infrastructure providers on an on- demand basis, so that cloud is subject to Quality of Service(QoS), Load Balance(LB) and other constraints which have direct effect on user consumption of resources controlled by cloud infrastructure [1] [2].

In Cloud, There are many tasks require to be executed by the available resources to achieve Minimal total timefor completion, Shortest response time, Effective utilization of resources etc, Because of these different intentions, we need to design, develop, propose a scheduling algorithm that is used by task scheduler to outperform appropriate allocation map of tasks on resources [4] [5].

2. MAX-MIN TASK SCHEDULING ALGORITHM

Max-min algorithm allocates task Ti on the resource Rj where large tasks have highest priority rather than smaller tasks [6]. For example, if we have one long task, the Max-min could execute many short tasks concurrently while executing large one. The total makespan, in this case is determined by the execution of long task. But if meta-tasks contains tasks have relatively different completion time and execution time, the makespan is not determined by one of submitted tasks. They try to minimize waiting time of short jobs through assigning large tasks to be executed by slower resources. On the other hand execute small tasks concurrently on fastest resource to finish large number of tasks during finalizing at least one large task on slower resource. Based on these cases, where meta-tasks contain homogeneous tasks of their completion and execution time, they proposed a substantial improvement of Max-min algorithm that leads to increase of Max-min efficiency. Proposed improvement increases the opportunity of concurrent execution of tasks on resources.

Algorithm:

For all submitted tasks in Meta-task; Ti

For all resources; Rj

Cij = Eij + rj

Find task Tk costs maximum execution time (Largest Task).

Assign task Tk to resource Rj which gives minimum completion time (Slowest resource).

Remove task Tk from Meta-tasks set.

Update rj for selected Rj.

Update Cij for all j.

While Meta-task not Empty

Find task Tk costs maximum completion time.

Assign task Tk to resource Rj which gives minimum execution time (Faster Resource).

Remove Task Tk form Meta-tasks set.

Update rj for Selected Rj.

Update Cij for all j.

The algorithm calculates the expected completion time of the submitted tasks on each resource. Then the task with the overall maximum expected execution time (Largest Task) is assigned to a resource that has the minimum overall completion time (Slowest Resource). Finally, this scheduled task is removed from meta-tasks and all calculated times are updated and then applying max-min algorithm on remaining tasks. Selecting task with maximum execution time leads to choose largest task should be executed. While selecting resource consuming minimum completion time means choosing slowest resource in the available resources. So allocation of the slowest resource to longest task allows availability of high speed resources for finishing other small tasks concurrently. Also, we achieve shortest make span of submitted tasks on available resources beside concurrency.

"Select task with the overall maximum expected execution time (Largest Task) then assign to be executed by resource with minimum expected completion time (Slowest Resource)".

3. ENHANCED TASK SCHEDULING ALGORITHM

Sometimes largest task is too large compared to other tasks in Mata-task, in that kind of case overall makespan is increased because too large task is executed by slowest resource first while other tasks are executed by faster resource OR when there is major difference among slowest and fastest resource in context of processing speed or bandwidth in that case largest task is

executed by slowest resource cause increasing in Makespan and load imbalance across resources.

Therefore, instead of selecting largest task if we select Average or Nearest greater than average task then overall makespan is reduced and also balance load across resources.

Algorithm:

For all submitted tasks in Meta-task; Ti

For all resources; Rj

Cij = Eij + rj

Find task Tk costs Average or nearest Greater than Average execution time.

Assign task Tk to resource Rj which gives minimum completion time (Slowest resource).

Remove task Tk from Meta-tasks set.

Update rj for selected Rj.

Update Cij for all j.

While Meta-task not Empty

Find task Tk costs maximum completion time.

Assign task Tk to resource Rj which gives minimum execution time (Faster Resource).

Remove Task Tk form Meta-tasks set.

Update rj for Selected Rj.

Update Cij for all j.

So in Enhanced Max-min, task selection scenario is changed, it is stated as "Select task with Average or Nearest greater than average execution time (Average or Nearest greater than average task) then assign to be executed by resource with minimum completion time (Slowest resource)".

4. PROPOSED WORK

The proposed work is simulated using Cloudsim which is a toolkit used for modeling and simulation. The deployment of the large scale applications is to be a great certain extent economical and easy by using clouds. At any intervals it is too expensive to measure applications performance in real cloud condition. So, cloudsim simulates the real cloud environment and gives the results that that has happened having real conditions.

Cloudsim architecture consists of basic entities such as Applications, Cloudlets, Cloud Broker, Virtual Machines, Hosts and Data Centers. These entities help to permits user to establish of a cloud computing environment and preceding the effectiveness of load balancing algorithms

The metrics used here in for the evaluation in terms of QoS are average Latency, Average turnaround time of all submitted tasks and total execution time of all submitted tasks. The simulation of the system is performed in Cloudlet with different number of Cloudlets varying from 200 to 2400.

Table 1 and Graph 1 shows the comparative analysis of average latency between proposed Work and Round Robin algorithm. Similarly, Table 2 and Graph 2 show the comparative analysis of average turnaround time and Table 3 and Graph 3 shows comparative analysis of total execution time of submitted tasks respectively.

Table 1: Comparative Analysis of Average Latency

S.No	No. of Task	Average Latency (in ms)	
		Proposed	Max-Min
1	200	2.92	6.54
2	400	7.65	13.104
3	800	13.25	25.21
4	1200	21.32	36.32
5	2400	45.23	78.36
6	Average	18.074	30.8232

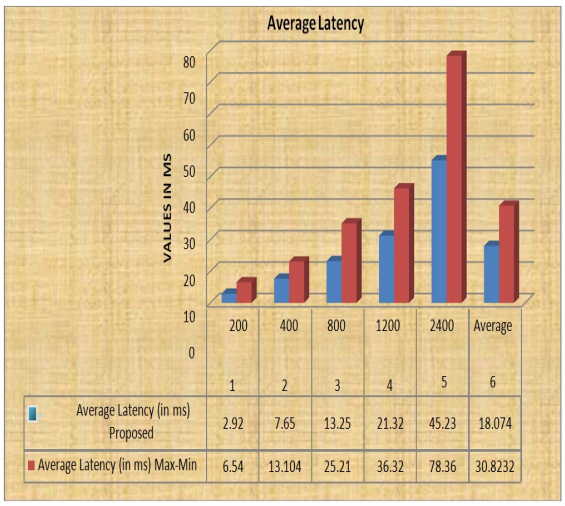

Figure 1: Comparative chart of average latency

Table 2: Comparative Analysis of Average Turnaround

S.No.	No. of Task	Average Turnaround Time (in ms)	
		Proposed	Max-Min
1	200	2.9	6.58
2	400	7.62	13.25
3	800	13.14	25.78
4	1200	21.3	36.82
5	2400	45.12	78.45
6	Average	18.016	32.176

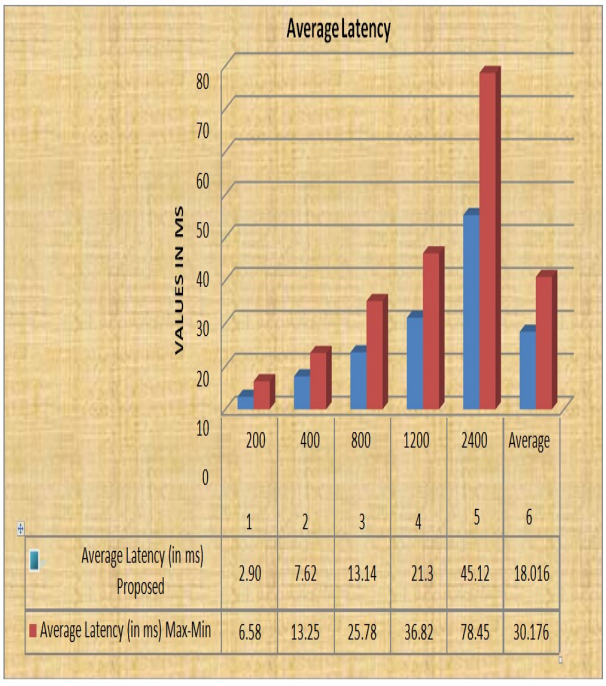

Figure 2: Comparative chart of average turnaround

Table 3: Comparative Analysis of Total Execution Time

S.No.	No. of Task	Total Execution Time (in ms)	
		Proposed	Max-Min
1	200	2	3
2	400	2	4
3	800	4	16
4	1200	4	24
5	2400	6	62
6	Average	3.6	21.8

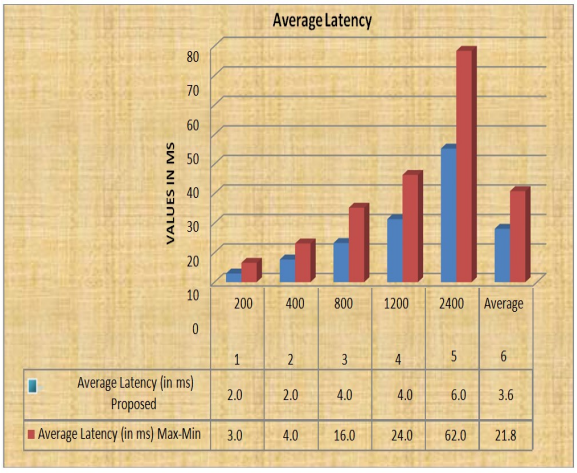

Figure 3: Comparative chart of total execution time

5. CONCLUSIONS

Task scheduling is main problem in cloud computing scenario. Proficient undertaking booking is vital for better use of assets. In this exploration, we have proposed proficient asset booking system for cloud environment. Effective asset booking system can help associations in decreasing force utilization and contribute straightforwardly to the organization's development and establishment's advancement. We have proposed Min-Min strategy which comes about into upgrading the different parameters to make the Cloud vitality productive. The proposed work is executed in java based test system CloudSim. The trial results demonstrates that the proposed strategies i.e. determination approach and low use host strategy results into profoundly advancing different parameters.

6. FUTURE SCOPE

Energy-efficient computing cannot be achieved without the integration between computer science and environmental science. Outlining server farms for the creating districts require vertically incorporated endeavors to drive key vitality proficient advancements in distributed computing, hardware (low power CPUs and frameworks). On the whole, these innovations address exceptionally noteworthy close term and long haul vitality challenges and natural issues. The green distributed computing in which distributed computing utilizing low power CPUs servers, and reestablishment vitality and most essential, which is closer to the end client. The future must have a methodology for a low vitality use server farms utilizing distributed computing that are controlled with renewable vitality choices streamlined by the very prescient calculations that depend on canny system of operators which monitor request and supply of both wellsprings of force and buyer of force from different appropriated areas. The future work to accomplish this, we should have more advanced occupation planning calculations in cloud computing environment.

REFERENCES:

[1] Salim Bitam, "Bees Life algorithms for job scheduling in cloud computing", International Conference on computing and Information Technology, 2012.

[2] Saeed Parsa and Reza Entezari-Maleki, "RASA: A New Grid Task Scheduling Algorithm", International Journal of Digital Content Technology and its Applications, Vol.3, pp. 91-99, 2009.

[3] O. M. Elzeki, M. Z. Reshad and M. A. Elsoud, "Improved Max-Min Algorithm in Cloud Computing", International Journal of Computer Applications (0975 – 8887).

[4] Braun, T.D., Siegel, H.J., Beck, N., Boloni, L.L, Maheswaran, M., Reuther, A.I., Robertson, J.P., et al. "A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems", Journal of Parallel and Distributed Computing, Vol. 61, No. 6, pp.810–837, 2001.

[5] He. X, X-He Sun, and Laszewski. G.V, "QoS Guided Minmin Heuristic for Grid Task Scheduling," Journal of Computer Science and Technology, Vol. 18, pp. 442-451, 2003.

[6] Etminani .K, and Naghibzadeh. M, "A Min-min Max-min Selective Algorithm for Grid Task Scheduling," The Third IEEE/IFIP International Conference on Internet, Uzbekistan, 2007.