International Journal of Application or Innovation in Engineering & Management (IJAIEM)
Web Site: www.ijaiem.org Email: editor@ijaiem.org, editorijaiem@gmail.com
Volume 2, Issue 4, April 2013 **ISSN 2319 - 4847**

Enhanced Max-min Task Scheduling Algorithm in Cloud Computing

Upendra Bhoi[1], Purvi N. Ramanuj[2]

[1]P.G.Student, Department of Computer Science & Technology, L.D.College of Engineering, Gujarat Technological University, Ahmedabad

[2]Assistant Professor, L.D.College of Engineering, Gujarat Technological University, Ahmedabad

ABSTRACT

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 Max-min task scheduling algorithm is proposed. The algorithm is built based on comprehensive study of the impact of Improved Max-min task scheduling algorithm in cloud computing. Improved Max-min is based on the expected execution time instead of completion time as a selection basis. Enhanced (Proposed) Max-min is also based on the expected execution time instead of completion time as a selection basis but the only difference is that Improved Max-min algorithm assign task with Maximum execution time (Largest Task) to resource produces Minimum completion time (Slowest Resource) while Enhanced Max-min assign task with average execution time (average or Nearest greater than average Task) to resource produces Minimum completion time (Slowest Resource).

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 time for 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. RESEARCH METHOD

This review aims at summarizing the current state of the art of various job scheduling techniques in cloud computing.

3. SOURCE OF INFORMATION

The Search was widely conducted in the following electronic sources to gain a broad perspective

- ScienceDirect (www.sciencedirect.com)
- ACM Digital Library (portal.acm.org)
- IEEE eXplore (ieeexplore.ieee.org)
- Springer LNCS (www.springer.com/lncs)

These sources cover the most relevant journals, conferences and workshop proceedings. The searches in the selected sources resulted in overlap among the papers, where the duplicates were excluded by manual filtering.

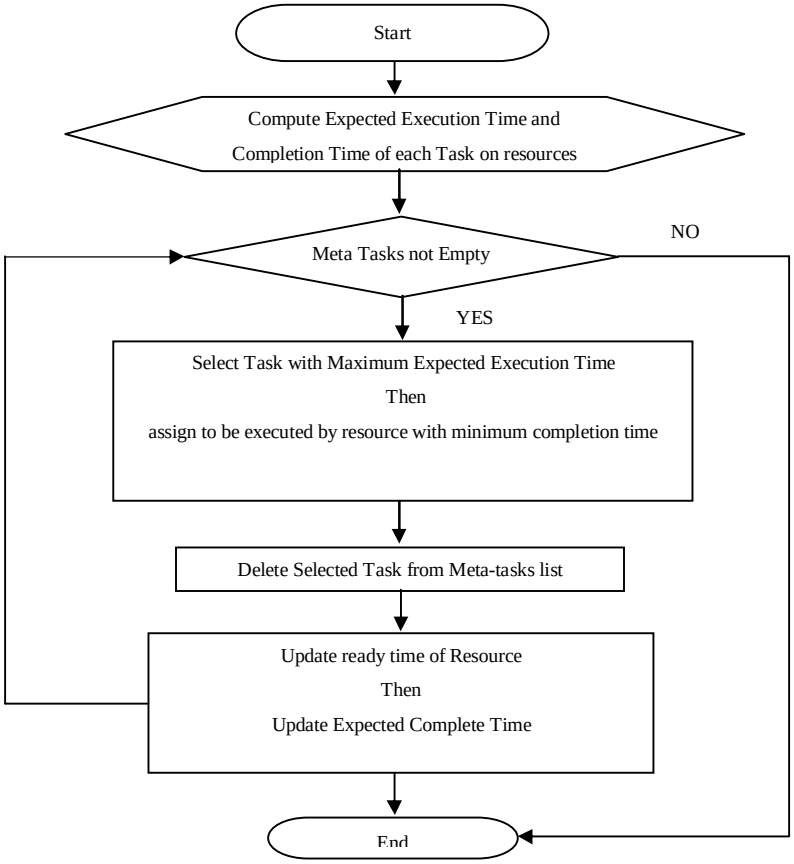
4. SEARCH CRITERIA

The initial search criteria included the titles (Job scheduling in cloud computing), (job scheduling techniques in cloud computing), (job scheduling in clouds) and (job scheduling in datacentres). The start year set to 2001, and the end year was 2012. Only papers written in English were included.

5. IMPROVED MAX-MIN TASK SCHEDULING ALGORITHM [3]

Max-min algorithm allocates task T_i on the resource R_j 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.

Flowchart:



Algorithm:

1. For all submitted tasks in Meta-task; T_i
   1.1. For all resources; R_j
      1.1.1. $C_{ij} = E_{ij} + r_j$
2. Find task T_k costs maximum execution time (Largest Task).
3. Assign task T_k to resource R_j which gives minimum completion time (Slowest resource).
4. Remove task T_k from Meta-tasks set.
5. Update r_j for selected R_j.
6. Update C_{ij} for all j.
7. While Meta-task not Empty
   7.1. Find task T_k costs maximum completion time.
   7.2. Assign task T_k to resource R_j which gives minimum execution time (Faster Resource).

7.3. Remove Task T_k form Meta-tasks set.
7.4. Update r_j for Selected R_j.
7.5. Update C_{ij} 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 makespan 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)".

6. ENHANCED MAX-MIN TASK SCHEDULING ALGORITHM (PROPOSED)

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:

1. For all submitted tasks in Meta-task; T_i
   1.1. For all resources; R_j
      1.1.1. $C_{ij} = E_{ij} + r_j$
2. Find task T_k costs Average or nearest Greater than Average execution time.
3. Assign task T_k to resource R_j which gives minimum completion time (Slowest resource).
4. Remove task T_k from Meta-tasks set.
5. Update r_j for selected R_j.
6. Update C_{ij} for all j.
7. While Meta-task not Empty
   7.1. Find task T_k costs maximum completion time.
   7.2. Assign task T_k to resource R_j which gives minimum execution time (Faster Resource).
   7.3. Remove Task T_k form Meta-tasks set.
   7.4. Update r_j for Selected R_j.
   7.5. Update C_{ij} 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)".

7. THEORETICAL ANALYSIS

Scenario 1: When there is major difference among slowest and fastest resource

Assume that Task scheduler has meta-tasks and resources as given below.

Table 1.1, represents the volume of instructions and data in tasks T_1 to T_5.

Task	Instruction Volume (MI)	Data Volume (Mb)
T_1	1400	98
T_2	1600	67
T_3	1200	45
T_4	800	56
T_5	1000	97

Table 1.1 Tasks Specifications

Table 1.2, represents processing speed and bandwidth of communication links of each resource.

Resource	Processing Speed (MIPS)	Bandwidth (MbPS)
R_1	100	150
R_2	400	50

Table 1.2 Resources Specifications

Using data given in Table 1.1 and Table 1.2, to calculate the expected execution time of the tasks on each of the resource.

Task	Resources	
	R1	R2
T_1	14	3.5
T_2	16	4
T_3	12	3
T_4	8	2
T_5	10	2.5

Table 1.3 Execution time of the tasks on each resource

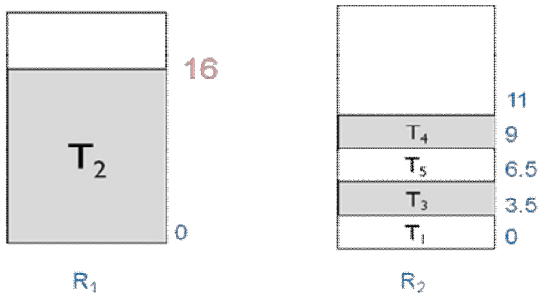


Figure 1.1 Gantt chart of Improved Max-min Algorithm (Scenario 1)

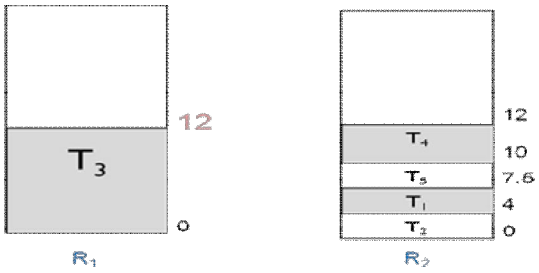


Figure 1.2 Gantt chart of Enhanced Max-min (Proposed) Algorithm (Scenario 1)

In above scenario, Enhanced Max-min achieves total makespan **12 second** while Improved Max-min achieves total makespan **16 second**.

Scenario 2: When largest task is too large compared to other tasks in meta-tasks

Table 2.1, represents the volume of instructions and data in tasks T1 to T7.

Task	Instruction Volume (MI)	Data Volume (Mb)
T_1	796	574
T_2	459	398
T_3	344	267
T_4	985	769
T_5	572	472
T_6	1500	296
T_7	5135	579

Table 2.1 Tasks Specifications

Table 2.2, represents processing speed and bandwidth of communication links of each resource.

Resource	Processing Speed (MIPS)	Bandwidth (MbPS)
R_1	100	150
R_2	200	100

Table 2.2 Resources Specifications

Using data given in Table 2.1 and Table 2.2, to calculate the expected execution time of the tasks on each of the resource.

Task	Resources	
	R1	R2
T_1	7.96	5.74
T_2	4.59	3.98
T_3	3.44	2.67
T_4	9.85	7.69
T_5	5.72	4.72
T_6	15	7.5
T_7	51.35	25.675

Table 2.3 Execution time of the tasks on each resource

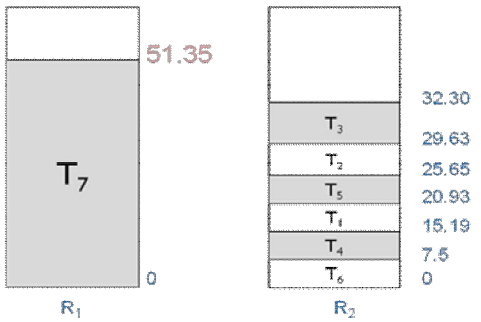


Figure 2.1 Gantt chart of Improved Max-min Algorithm (Scenario 2)

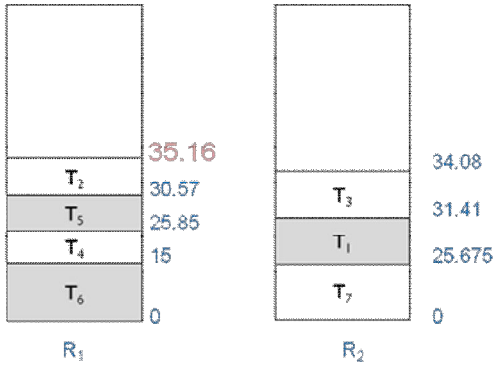


Figure 2.2 Gantt chart of Enhanced Max-min (Proposed) Algorithm (Scenario 2)

In above scenario, Enhanced Max-min achieves total makespan **35.16 second** while Improved Max-min achieves total makespan **51.35 second**.

8. CONCLUSIONS AND FUTURE WORKS

When we schedule tasks using improved max-min task scheduling technique then in case, 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 too slow resource cause increasing in Makespan and load imbalance across resources.
Therefore, a unique modification of Improved Max-min task scheduling algorithm is proposed. The algorithm is built based on comprehensive study of the impact of Improved Max-min task scheduling algorithm in cloud computing. Improved Max-min algorithm assign task with Maximum execution time (Largest Task) to resource produces Minimum completion time (Slowest Resource) while Enhanced Max-min assign task with average execution time (average or Nearest greater than average Task) to resource produces Minimum completion time (Slowest Resource). This reduces overall makespan and balance load across resources.
For simulation, we will use CloudSim which is java based simulation toolkit that enables modelling, simulation and experimenting on designing cloud computing infrastructures.

ACKNOWLEDGEMENT

Upendra Bhoi would like to thank to his thesis guide Prof. Purvi N Ramanuj for her great effort and instructive comments in this paper work. Lastly, I wish to thank to all those who helped me during the lifetime of my research work.

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.

AUTHOR

Upendra Bhoi received the B.E. degree in Computer Engineering from Atmiya Institute of Technology and Science in 2010. He is receiving M.E. degree in Computer Science and Technology from Gujarat Technological University, Ahmedabad.