Performance analysis of Dual objective load shared minimum execution time grid task scheduling algorithm

M. Hemamalini

https://doi.org/10.56343/STET.116.015.001.007
www.stetjournals.com

Article History
Received: 29.09.2020
Revised and Accepted : 15.12.2020
Published: 24.09.2021

Abstract

Grid computing is a conventional technology to integrate large scale distributed resource sharing. Task scheduling is an essential problem for achieving high performance in grid computing systems. Many parameters like execution time, memory capacity, network bandwidth, communication cost etc affect the task scheduling in Grid environment. An effective and efficient scheduling algorithm is needed to attain the promising potentials of significant distributed resources and to improve the overall throughput of the Grid environment. Minimum Execution Time scheduling algorithm fails to produce a load balanced schedule as well as it considers only the execution time of tasks for scheduling. Dual Objective Load shared Minimum Execution Time algorithm considers the Quality of Service (QoS) factor such as memory requirement of task and the execution time of the task to make the algorithm efficient in real time environment. The performance of Dual Objective Load shared Minimum Execution Time [DOLSMET] analyzed with traditional Minimum Execution time scheduling algorithm. The result shows that Dual Objective Load shared Minimum Execution Time algorithm produces an efficient schedule that balances the load as well as satisfies the memory requirements of tasks.

Key words: Grid Computing, load balancing, memory requirement, Minimum Execution Time algorithm, Task Scheduling.

INTRODUCTION

Grid computing is an essential technology mainly used for distributed environment. Grid is defined by Ian Foster as flexible, secure, large scale resource sharing among dynamic collection of individuals and institutions (Ian Foster, 2001). Mixed-machine heterogeneous computing environments (Braun *et al.*, 2001) are a collection of heterogeneous high-performance machines interconnected with high-speed links. They are used to solve a computationally intensive application that needs different computing environments. Computational Grids (Chapman *et al.*,2007) are considered as the next generation of distributed system. Computational grid provides the best solution for business problems, scientific and engineering applications which needs huge amount of resources. Many researchers and developers focus on the challenging issues like scheduling and resource management in the grid computing era. Scheduling (Siriluck Lorpunmanee *et al.,* 2007) is the most recent topic in the Grid Scenario. Effective and efficient task scheduling algorithm is needed to achieve high performance in grid environments. The main goal of grid task scheduling is to increase resource utilization and reduce the makespan.

The success of the grid computing is how effectively it schedules the tasks with available resources. Grid system allocates the tasks to the available resources based on user's requirement. Heterogeneous computing environment utilizes the different high performance resources to perform massive application that have different computational requirements institutions (Ian Foster, 2001). The matching of tasks to resources and scheduling the execution order of these tasks is referred to as mapping. The general problem of mapping tasks to resources in an heterogeneous environment has been shown to be NP- Complete (Fernandez-Baca, 1989).

Metatask can be defined as a collection of independent tasks with no intertask data dependencies. The main objective of this mapping is to reduce the total execution time of the metatask. It is also assumed that each resource executes a single task at a time based on the order in which the tasks are

M. Hemamalini

email: *maliniavcce@gmail.com*

Assistant Professor, Department of Computer Science, A.V.C. College (Autonomous), Mannampandal, Mayiladuthurai - 614016, Tamil Nadu, South India.

P - ISSN 0973 - 9157
E - ISSN 2393 - 9249

assigned. The size of the metatask and the available number of resources are known priori (Braun *et al*., 2001). These algorithms schedules the tasks based on execution and completion time of each task on each available resources. There is no best task scheduling algorithm for grid systems. An alternative way is to choose an algorithm based on the characteristics of task, resource and network and memory requirement. The paper is organized as follows. The existing algorithms for scheduling are discussed in Section 2. Problem description is given in Section 3. The Dual Objective Load Shared Minimum Execution Time scheduling algorithm is presented in Section 4. In Section 5 Experimental results are presented. Finally, conclusion and directions for future work are presented in Section 6.

RELATED WORKS

The major issue in the Grid computing system is the development of efficient techniques for the mapping of the tasks to resources, and to minimize the makespan of the program (Selvi and Amalarethinam, 2012). Traditional job scheduling algorithms like First Come First Serve, Shortest Job First, etc., may not be suitable for the dynamic environment in grids (Kokilavani and George Amalarethinam, 2011) due to the heavy heterogeneity nature in the Grid environment,. Thus the researchers are building many scheduling algorithms specifically for the grid environments. In this section we have reviewed a set of algorithms designed for meta-task scheduling in grid environment.

Load balancing algorithm distributes the load among all the available resources. The algorithm tries to increase the utilization of resources with light load and freeing the resources with heavy load. These algorithms are mainly used to minimize the makespan with the effective utilization of resources.

Braun *et al* (2001) have studied the performance of eleven grid task scheduling algorithms. They have also provided a simulation basis for researchers to test the grid task scheduling algorithms. Their result shows Genetic Algorithm (GA) outperforms the other algorithms. Min-Min algorithm performs next to GA and the rate of improvement is very small. Braun *et al* have proposed the scheduling algorithms such as Opportunistic Load Balancing (OLB), Minimum Execution Time (MET), Minimum Completion Time (MCT), Min-Min and Max-min algorithms.

Opportunistic Load Balancing (OLB) assigns the jobs in a random order in the next available resource without considering the execution time of the jobs on those resources. Thus it produces poor makespan and a load balanced schedule.

Minimum Execution Time (MET) grid task scheduling algorithm finds the task which has minimum execution time and assigns the task to the resource based on first come first served basis. The main drawback of this algorithm is severe load imbalance. It does not consider the availability of the resource and its load.

MET algorithm finds the task which has minimum execution time and assigns the task to the resource based on first come first served basis (Hemamalini, 2012). In a Balanced Minimum Execution Time scheduling algorithm, rescheduling can be performed based on maximum completion time of the task. Thus it increases the resource utilization and load is balanced. The main drawbacks of this algorithm are finding priority of job which is a tedious one and higher turnaround time.

Optimal Resource Constraint Scheduling algorithm distributes the task among the available processor based on processor capability. It is an efficient load balanced task scheduling algorithms which reduces the turnaround time and average waiting time. It is suitable for more number of jobs and avoids starvation problem.

Kokilavani *et al* (2012) proposed Load Balanced Min-Min scheduling algorithm which produces better results than min- min scheduling algorithm. It reduces the makespan and balance the load. The response time is improved and load balancing is achieved efficiently. This algorithm applies the min-min grid scheduling algorithm in the first phase and rescheduling takes place based on maximum execution time (Kokilavani and George Amalarethinam, 2011).

He *et al.*(2003) have proposed a new algorithm QoS guided Min-Min Grid Task scheduling algorithm which is based on the conventional Min-Min algorithm. This algorithm schedules tasks requiring high bandwidth before the others. Therefore, if the bandwidth required by different tasks varies highly, the QoS guided Min-Min algorithm provides better results than the Min-Min algorithm (He *et al.*,2003).

Salehi *et al.*(2012) have proposed preemption-aware scheduling policy for a virtualized multi-cluster Grid that distributes Grid requests among different clusters, in a way that the number of preemptions minimizes. The proposed policy is based on the stochastic analysis of routing in parallel non-observable queues. This policy is not dependent to the availability information of the clusters, and does not impose any overhead on the system (Salehi *et al*., 2012).

Chauhan and Joshi (2010) have proposed a QoS Guided Weighted Mean Time Min-Min Max-Min

selective heuristic, for QoS based task scheduling. In fact, this work adds QoS parameters to weighted Mean Time Min-Min Max-Min Selective (WMTS) heuristic algorithm. The Weighted Mean Time Min-Min Max-Min Selective heuristic considers the performance of resources, and calls this performance as weight of resource. It uses the merits and demerits of Min-Min and Max-Min heuristics for scheduling. It includes QoS parameter and network bandwidth in it. In this method, first the meta-tasks divided into high and low QoS groups, that the high QoS group contains the tasks with high requirement QoS, and the other one includes the tasks with low QoS requirements. It schedules the tasks from high QoS group first and afterward, tasks from low QoS group (Chauhan and Joshi, 2010).

A load balancing algorithm tries to move the load from heavily loaded resources to idle resources. At the same time it aims to minimize the makespan. Among all the algorithms available, Minimum Execution Time scheduling algorithm is considered to be simple. But it does not achieve load balancing. In this paper we propose a Dual Objective Load shared Minimum Execution Time scheduling Algorithm which considers the memory requirement of tasks and Expected Time to Compute (ETC) matrix for best scheduling.

PROBLEM STATEMENT

Task scheduling is one of the NP-Complete problems. Let T1, T2, T3, T4 and T5 are collection of independent tasks. The meta task refers the tasks which have no dependency among each others. Each task is assigned to a resource based on the order in which the tasks are arrived.

The input to this algorithm is size of the meta task and number of resources. The expected execution time for each task on each resource is known prior to execution of the task. The Expected Time to Compute Matrix ETC (Ti, Rj) gives details of the expected execution time of a task on a particular resource which is extensively used in meta-task scheduling, where Ti represents meta- task and Rj represents Resource Set. The Problem can be defined as follows:

Let task set Ti= T1, T2, T3, T4, …, Tn.

Let Resource Set Ri = R1, R2, R3, R4,……..,Rn.

An ETC matrix is a n x m matrix in which m is the number of resources available and n is the number of tasks to be scheduled. Each row of ETC matrix contains the expected execution time of a task on each machine. Similarly each column of the matrix contains the execution time taken by a machine for each task. The expected time taken by task Ti on machine Mj is represented as ETij and the completion time of a task i on machine j is represented as CTij.

CTij is calculated using equation 1.

$$CTij = ETij + Rj \quad --(1)$$

The makespan can be calculated as follows:

Makespan = $\max(CT\ (Ti, Rj)$ $--(2)$

Where ETij = Execution time of Task i on Resource j.

CT = Completion Time

Rj = Ready Time of Resource j.

Grid task scheduling is used to find the acceptable solution with fewer costs. Grid Task scheduling is one of the NP-Complete Problem. The goal of scheduler is to minimize the makespan.

PROPOSED METHOD

The limitation of Minimum Execution Time scheduling algorithm is that it does not produce a load balanced schedule. After scheduling, the task could not be run on the assigned machine due to the resources capacity is lesser than the requirement of the task. The Dual Objective Load shared minimum Execution Time scheduling algorithm considers one of the essential parameters memory requirements and processor time. In DOLSMET the scheduling is done in two steps. The resource with memory capacity that matches the task requirements is added in a resource list RL. This resource list is created for each task. In the second step, load balanced Minimum Execution Time scheduling algorithm is executed on the selected resources in resource list RL.

Thus the scheduler searches for available resources after getting a batch of jobs. From the available resources it compares the memory requirement of task and the capacity or resources. The scheduler forms a resource list for each task in which it adds the resources whose capacity is greater than or equal to the requirement of the task. Dual Objective Load shared Minimum Execution Time Algorithm for Meta Task Scheduling in Grid Computing list for each task which has minimum completion time and produces a load balanced schedule. The procedure for the Dual Objective Load shared Minimum Execution algorithm is given below.

Algorithm for Dual Objective Load Shared

Minimum Execution Time Grid task scheduling

The DOLSMET applies a selection policy based on memory requirements of the task in the first step. It takes the results from the first step and then applies the selection policy based on processor time in the second step. If a resource is heavily loaded, it

```
Procedure DOLSMET
Read the task requirements and the resource
characteristics
For all tasks
Find the resource list (RL) which has
   memory size >= required memory size of task
End for
For all tasks
Find Minimum Execution Time of task from
   RL and place it in Task Queue Q
End For
While Q not empty
Schedule the task Ti in resource Rj
   which produces MCT where Rj in RL
Remove Task Ti from Q
 End While
For all resources
Find makespan=max(Cj)
End for
Find the resource with maximum makespan
For all resources R
Compute makespan = max(CT(R))
End for
For all resources  For all tasks
find the task Ti that has minimum Execution Time  in Rj
Find the MCT of task Ti
If MCT < makespan
Reschedule the task Ti to Rj that produces MCT
  provided Rj in RL.
Update the ready time of both resources
End if
End for
```

Table 1. Resource Characteristics

Resource	Available RAM
R1	160
R2	140
R3	170

Table 2. Execution Time and Memory Requirement of the task

Task	Memory Requirement of	Execution Time in milliseconds		
		R1	R2	R3
T1	120	110	85	56
T2	130	120	81	101
T3	160	123	96	120
T4	150	35	111	159
T5	130	74	71	48
T6	125	112	67	54

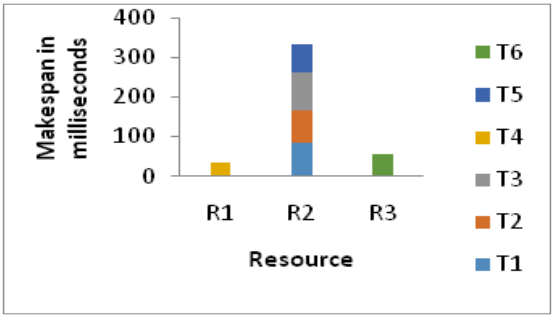

Fig. 1. The makespan produced by Minimum Execution Time scheduling algorithm

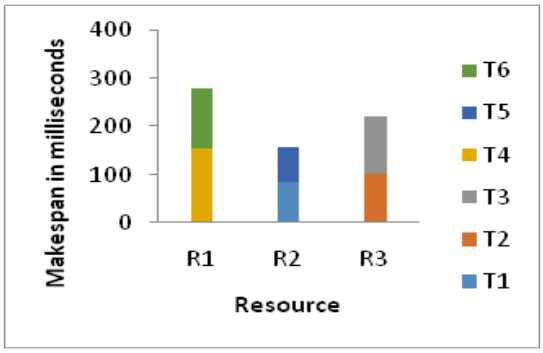

Fig. 2. The makespan produced by DOLSMET scheduling algorithm

Table 3. Comparison of MET and DOLSMET Algorithms

Problem Set	MET (Makespan in millisecond)	DOLSMET (Makespan in millisecond)
P1	333	278
P2	209	181
P3	215	195
P4	245	201
P5	205	190

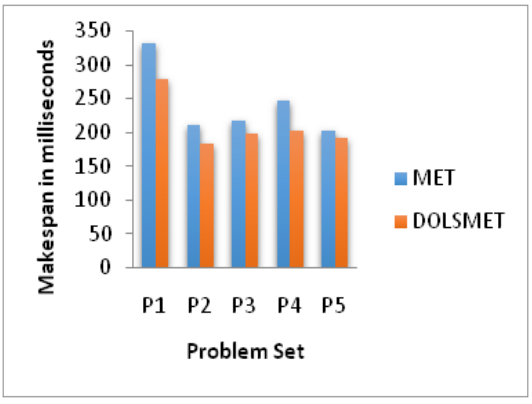

Fig. 3. Graphical Representation to compare the performances of DOLSMET and MET

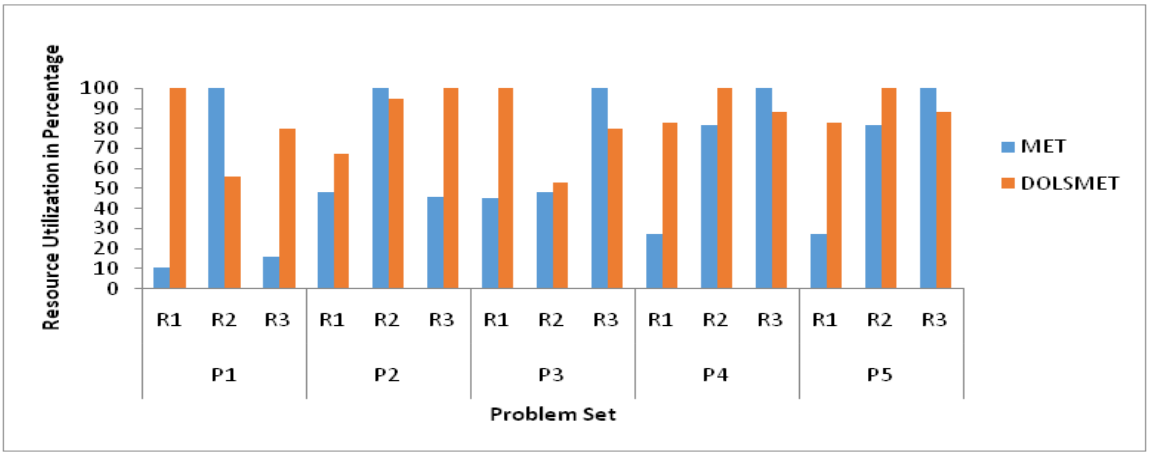

Fig. 4. Resource Utilization in Percentage of MET and DOLSMET

Table 4. Resource Utilization Rate in Percentage of MET and DOLSMET

Problem Set	Resource	MET	DOLSMET
P1	R1	10.5	100
	R2	100	56.1
	R3	16.2	79.5
P2	R1	47.9	67.2
	R2	100	94.5
	R3	45.5	100
P3	R1	45.2	100
	R2	48.4	53
	R3	100	79.8
P4	R1	27.5	82.7
	R2	81.8	100
	R3	100	88.1
P5	R1	27.5	82.7
	R2	81.7	100
	R3	100	88.1

reschedules the tasks to get a load balanced schedule. This algorithm considers the resources in a particular task's resource list for rescheduling also.

EXPERIMENTS AND RESULTS:

Consider a heterogeneous grid environment with three resources R1 and R2 and independent Task group M with six tasks T1, T2, T3, T4,T5 and T6. The grid scheduler schedules all the available task on the available resource R1, R2 and R3. Table 1 represents Resource characteristics. Table 2 shows the Execution Time and Memory Requirement of the task

Scheduling of the tasks to resources based on Minimum Execution Time as given in Algorithm. MET chooses the minimum execution time. The makespan produced by MET is 333 milliseconds. Since the Minimum Execution Time scheduling algorithm does not consider the memory requirements of a task, the tasks may suffer during scheduling because of the unavailability of the memory during run time. Figure 1 shows the makespan produced by Minimum Execution Time scheduling algorithm.

Thus the DOLSMET scheduling algorithm contains fault tolerance mechanism by identifying the problem before scheduling. The DOLSMET scheduling algorithm schedules the task based on memory requirement of the task and processor time as well as it balances the load. The makespan produced by DOLSMET scheduling algorithm is 278 which is lesser than the makespan produced by MET algorithm. The results are analyzed using graphs shown in Figure2. All the results show that DOLSMET outperforms the traditional Minimum Execution Time scheduling algorithm.

Let us take the example problems having both task and resource heterogeneity and executes for both MET and the Dual Objective Load Shared Minimum Execution Time Scheduling Algorithm.. The Table 3 shows the results (in milliseconds) of both algorithms.

The results are plotted in a graph (Fig.3). The Dual Objective Load shared Minimum Execution Time Scheduling algorithm produces less makespan than MET scheduling algorithm. The Figure 3 shows Dual Objective Load shared Minimum Execution Time Scheduling algorithm outperforms MET scheduling algorithm.

Table 4 shows the resource utilization rate in Percentage for five different Problem sets P1, P2, P3, P4 and P5. The Dual Objective Load shared Minimum Execution Time Scheduling algorithm balances the load and reduces the makespan by using unutilized resource in the second phase. Table 4 shows that

Memory Constrained LSMET efficiently utilizes all the available resource. Resource Utilization can be calculated using the formula.

$$UR = Ti * 100 / TQRU \quad --(3)$$

$$TQRU = \sum_{i=1}^{n} CT \quad -(4)$$

TQRU = Total Quantity of Resource Used.

Ti = Meta task.

UR = Usage of Resource.

CT = Completion Time of Task.

The resource utilization percentage is shown in Figure 4 for five different problem set P1, P2 , P3, P4 and P5. From this figure we can observe that Dual Objective Load shared Minimum Execution Time Scheduling algorithm uses the maximum amount of resources while reducing the makespan obtained from MET algorithm.

The Figure clearly shows that for all types of heterogeneity values the DOLSMET algorithm gives reduced makespan. This result is achieved by balancing the load and using the resources efficiently.

CONCLUSION

Scheduling a task on the appropriate resource is difficult in a distributed environment like Grid. For static meta-task scheduling algorithms, the tasks may suffer problem during execution because of the various parameters. Memory requirement of the task is one of the important parameters to be considered during execution. The MET algorithm considers only the execution time of the tasks for scheduling and it does not balance the load. In this paper, Dual Objective Load Shared Minimum Execution Time scheduling algorithm is analyzed based on the traditional MET algorithm. The DOLSMET algorithm produces better results for heterogeneity of tasks and resources. The DOLSMET algorithm tries to balance the load as well as provides fault tolerance mechanism by considering the memory requirement and processor time as a parameter for scheduling the tasks. The results show that DOLSMET outperforms the traditional MET algorithm for all cases. The work may be further extended by considering other factors like network bandwidth, communication delay and so on in the future.

REFERENCES

Braun. T.D., H.J Siegel., N.Beck,., L.L.Boloni, M.Maheswaran, A.I.Reuther, J.P.Robertson, M. D. Theys, and B.Yao. 2001. A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. *J. Parallel and Distr. Com.*, 61:810–837.

Chapman. C, M.Musolesi, W.Emmerich, C.Mascolo. 2007. Predictive Resource Scheduling in Computational Grids in parallel and Distributed Processing Symposium, *IEEE International*, Vol. 26:1 – 10.

Chauhan, S.S. and R.C. Joshi. 2010. QoS guided heuristic algorithms for grid task scheduling. 2010. *Int. J. Comput. Appl.* 2(9):24-31.

Fernandez-Baca. D. 1989. Allocating modules to processors in a distributed system, *IEEE Trans. Software Engg.* 15(11). P.1427-1436.
https://doi.org/10.1109/32.41334

He, X., X. Sun and G. Laszewski. 2003. A QoS guided min-min heuristic for grid task scheduling. *J.Comp. Sci. and Technol.* 18(4): 442-451.

Hemamalini. M and Dr.M.V.Srinath. 2014. Balanced Minimum Execution Time Grid task scheduling Algorithm in a Heterogeneous Distributed Environment. *Proceedings of IEEE International Conference on Advances in Engineering and Technology*, 2014.

M.Hemamalini. 2012. Review on Grid Task scheduling Algorithm in Heterogeneous Distributed Environment", *Int. J. Comput. Appl.* 40(2):.24-30.

Ian Foster, Carl Kesselman, Steven Tuecke.2001. "The Anatomy of the Grid Enabling Scalable Virtual Organizations", *Int. J. High. Perform. Comput. Appl.*15:200-222

Kokilavani, T. Dr. George Amalarethinam, D.I. 2011. Load Balanced Min- Min Algorithm for static Meta–Task Scheduling in Grid Computing. *Int. J. Comput. Appl.*, 20(2):43-49

Kokilavani. T and George Amalarethinam.D.I, 2012. An Ant Colony Optimization Based Load Sharing Technique for Meta Task Scheduling in Grid Computing", Proceedings of the Second *International Conference on Advances in Computing and Information Technology, AISC 177*, Springer, Vol. 2, pp. 395 – 404.

Salehi, M.A., B. Javadi and R. Buyya. 2012. QoS and preemption aware scheduling in federated and virtualized Grid computing environments. *J. Parallel Distr. Com.* 72: 231–245

Selvi, F.K.M. and D.I.G. Amalarethinam. 2012. Task Scheduling Algorithms to Minimize Makespan in Computational Grid Environment. Ph.D. Dissertation, Bharathidasan University, Tiruchirappalli, India.

Siriluck Lorpunmanee, Mohd Noor Sap, Abdul Hanan Abdullah, and Chai Chompoo-inwai. 2007. An Ant Colony Optimization for Dynamic Job Scheduling in Grid Environment", *World Acad. Sci. Eng. Technol.* 29:314- 321.