Heuristic Algorithm for Balancing Load in Grid Task Scheduling

V.Vasudevan, PhD.
Department of Information Technology
Kalasalingam University, Krishnankoil

R.Vijayalakshmi
Department of Computer Applications
Kalasalingam University, Krishnankoil

ABSTRACT
Grid Computing is an important field that focuses on resource sharing. Grid Computing provides a secure, control and flexible resource access environment in a distributed network. One of the most critical issues in Grid Computing is efficient scheduling of the tasks. The main aim of Grid scheduling is to map the tasks onto the available processors and order their execution. Due to the dynamism and heterogeneity of the grid, an efficient scheduling algorithm that minimizes makespan with maximum resource is necessary. Efficient scheduling of jobs to the available grid resources makes effective utilization of the grid environment. Heuristic algorithms can be used for solving task scheduling problems, since it is shown to be NP-Complete. Efficiency of scheduling algorithms can be evaluated using the two important criteria makespan and resource utilization. A heuristic task scheduling algorithm that satisfies load balancing of resources on a grid environment is presented in this paper. This algorithm schedules the tasks which reduces the makespan of the jobs and increase the utilization of resources. The new heuristic task scheduling is compared with other traditional heuristics and the results are shown to predict that the new algorithm outperforms the other.

Keywords
Grid Computing, Task Scheduling, Load Balancing, Heuristics, Makespan, Resource Utilization.

1. INTRODUCTION
Grid computing tries to bring under one definitional umbrella all the work being done in high-performance, cluster, peer-to-peer and internet computing. Grid computing has the ability to form virtual, collaborative organization that share applications and data in an open heterogeneous server environment in order to work on common problems. It provides a hardware and software infrastructure that helps dependable, consistent, pervasive and inexpensive access to computational resources.

Grid computing is considered as the best solution for solving most complex, scientific, and engineering and business problems that need large amount of resources for execution [1]. Scheduling, performance prediction and resource management are some of the challenging issues in the Grid environment [2]. Difficulty in grid scheduling is due to its diverse operating system, architecture and resources.

Current grid computing scenario considers scheduling as an important issue [1]. An optimal resource allocation that minimizes the schedule length of jobs and makes effective resource utilization is difficult to find [3]. Tasks in a grid environment can be divided into independent tasks that do not communicate with the other and dependent tasks that communicate with other. In this paper, the scheduling of independent tasks is considered and makespan and resource utilization are the criterions assumed.

2. RELATED WORKS
Several heuristic algorithms have been proposed to schedule tasks in grid computing environment. Heuristics task scheduling strategies are used for optimal task scheduling. This section reviews the commonly used algorithms Minimum Completion Time (MCT), Minimum Execution Time (MET), Min-Min and Max-Min that schedule meta-task (MT) to a set of resources.

MET regardless of the machine availability time, assigns each task to the machine with the minimum expected execution time in arbitrary order. The objective of MET is to allocate each task to its best resource [4]. This algorithm cause severe load imbalance across the resources even though it improves makespan to some extent.

Minimum Completion Time Algorithm assigns a task to the resource which has minimum completion time for it [4]. The task assigned to a resource is removed from the set of tasks and the completion time for all the remaining tasks are updated by adding the ready time and execution time of the resource. The process is repeated until all the tasks are mapped. This algorithm considers only one task at a time. Some tasks are assigned to the machines that do not have minimum execution time for them, since it assigns tasks in an arbitrary order.

Min-Min algorithm uses minimum completion time as a metric [4]. It starts with a set of all unmapped tasks. The machine that has the minimum completion time for all jobs is selected. Then the job with the overall minimum completion time is selected and mapped to that resource. The task assigned to the resource is deleted from the set of tasks, and the ready time of the resource is updated. This process is repeated until all the unmapped tasks are assigned [5]. Compared to MCT, this algorithm considers all jobs at a time and produces a better makespan. The drawback of this heuristics is that it chooses smaller tasks first which makes use of resource with high computational power. Hence, the schedule produced by Min-Min is not optimal when the smaller tasks exceed the larger tasks. Though it aims to minimize makespan, it results in unbalanced load and poor utilization of resources.

Max-Min heuristic is very similar to Min-Min and its metric is also minimum completion time [4]. It begins with a set of all unmapped tasks. It computes the completion time for each task on every machine. The machine with the minimum completion time for each task is selected. The task with maximum completion time is finally mapped to the selected machine. The task assigned is removed from the set of tasks and the completion time of all the tasks is updated. This continues until all the tasks are updated [6]. This heuristics

provides better performance than Min-Min when the number of small tasks is larger than the number of larger tasks.

3. PROBLEM DEFINITION

The most important part of grid resource management system is task scheduling. The scheduler selects the appropriate resource to run the task whenever it receives a request from a task. A unique resource is selected to complete the tasks since they are independent. This paper proposes a grid task scheduling algorithm that assumes the following assumptions:

Tasks are independent.
Tasks that have no communication among each other are considered.
No deadlines or priorities associate with the tasks.
Perform static mapping of process in batch mode.
Number of resources and number of tasks are known.
One task is executed on a machine at a time in FCFS order.

Static heuristics algorithms estimates the expected execution time for each task on each machine using Expected Time to Compute matrix [3] where ETC (t_i,r_j) is the estimated execution time of task i on resource j. Using ETC matrix model, the grid task scheduling problem can be defined as follows:

Let the Group of tasks submitted to the scheduler T = t1, t2, t3,…….tn and set of resources available at the time of task arrival R = r1, r2, r3,……rk.

Heuristic Task Scheduling Algorithm is designed to work for the above stated problem with the aim of minimizing makespan and balancing resource utilization.

3.1 Performance Metrics

The metrics used for evaluating the performance of the grid task scheduling algorithm are makespan and resource utilization.

3.1.1 Makespan

Makespan is defined as the maximum completion time of resources [3]. Makespan is calculated as follows,

$$\textbf{Makespan} = \max (CT (t_i, m_j))$$

$$CT_{ij} = E_{ij} + R_j$$

R_j – Ready time of resource j after completing the assigned jobs.

CT – Completion time of machines

E_{ij} – Expected Execution Time of job i on resource j.

3.1.2 Resource Utilization

Resource Utilization is defined as the amount of resources busy in executing tasks [3]. Resource utilization is calculated using the following formula.

$$\textbf{Resource Utilization} = M_i * 100 / TARU$$

$$\textbf{Total Amount of resource Used} = \sum_{i=1}^{n} CT_i$$

3.2 HTSA

Figure 1 presents the proposed algorithm. This algorithm first computes the completion time of resources from the given Expected Execution time of task on a resource. It then chooses the minimum execution time task and assigns the task to the resource that produces minimum time for execution. This process is continued till all the tasks are assigned by matching the minimum execution time. This algorithm balances the load by selecting the resource that produces the nearest makespan provided by the previous cycle. It computes the minimum execution time of task on the resource. The makespan produced in the first cycle is compared with the maximum completion time. If it is less than the makespan, then the task is rescheduled by selecting the resource that has the nearest makespan and the ready time of the resource is updated. The process is repeated till all the tasks are assigned for a particular resource. Hence HTSA produce a schedule that balances load optimally on all the resource without keeping a resource idle or with minimum load.

```
For all tasks in the set T
  For all resources
    Compute Cij = Eij+Rj
Do until all tasks in the set are mapped
   Find the earliest completion time for each task and the
   resource that obtains it
     Find the task Ti with the minimum earliest completion
     time
     Assign task Ti to the resource that gives the earliest
     completion time
     Delete task from the set Ti
     Update ready time of resource Rj
     Update Cij for all i
End do
Arrange the resources in the order of completion time
For all resources R
     Compute makespan = max(CT(R))
 End for
 For all resources
   For all tasks
       Find the task Ti that has minimum ET in Rj
       Find the Maximum Completion Time of task Ti
       If Maximum Completion Time < makespan
         Select the resource that has the nearest makespan
         Reschedule task Ti to the resource that produces  it
         Update the ready time of both resources
       End if
   End for
```

Figure 1. HTSA

4. ILLUSTRATIVE EXAMPLE

Consider a grid environment with two resources R1, R2 and a meta-task group M with four tasks T1, T2, T3 and T4.All the tasks are supposed to be scheduled to the machines R1 and R2.The ETC matrix for the problem statement is illustrated in Table 1.

Table 1. Expected Execution Time of Tasks

Tasks	Resources	
	R1	R2
T1	2	3
T2	4	8
T3	5	7
T4	4	3

Figure 2 shows the performance of the heuristic algorithms. Min-Min and Max-Min achieves a makespan of 10. Similarly the makespan achieved is 11 and 10 for MET and MCT

respectively. The results are shown in Figure 2. HTSA achieves a minimum makespan of 9 when compared to other heuristics.

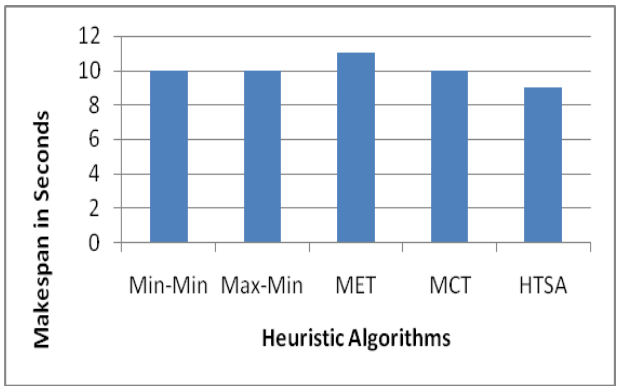

Figure 2. Comparison of Makespan among heuristics

In this example, since, Min-Min prefers minimum completion time, the resource utilization for R1 is 60% and R2 is 100. While using Max-Min scheduling, 100% resource utilization is made on R1 and 70% in R2. MET uses 100% of R1 and 27.27% of R2 and creates a more unbalanced utilization. MCT uses 60% of R1 and 100% of R2 whereas HTSA produces better load balancing with 100% of R1 and 88.88% of R2. Figure 3 shows the resource utilization chart.

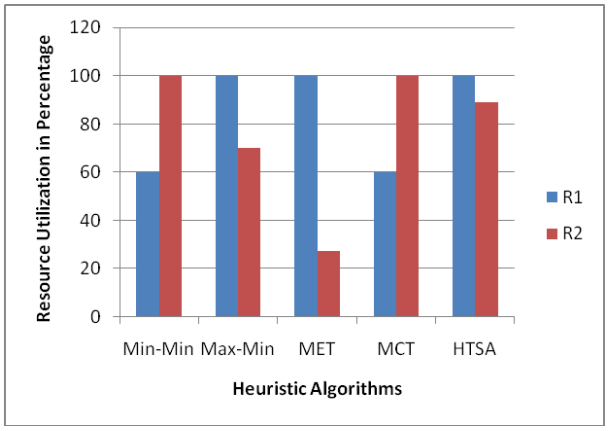

Figure 3. Comparative Analysis of Resource Utilization

5. RESULTS AND DISCUSSION

Different problem sets from various literatures are taken and executed using coding developed in C++ for HTSA and other heuristic algorithms. The results obtained are tabulated in Table 2.

Table 2. Comparison of heuristics – Makespan in Seconds

Problem Set	Min-Min	Max-Min	MET	MCT	HTSA
P1	8	6	8	8	6
P2	11	10	16	13	9
P3	30	30	45	30	20

Figure 4 shows the comparison of traditional heuristics and our proposed algorithm. It shows that the makespan obtained by HTSA is less compared to other heuristic algorithms Min-Min, Max-Min, MCT and MET.

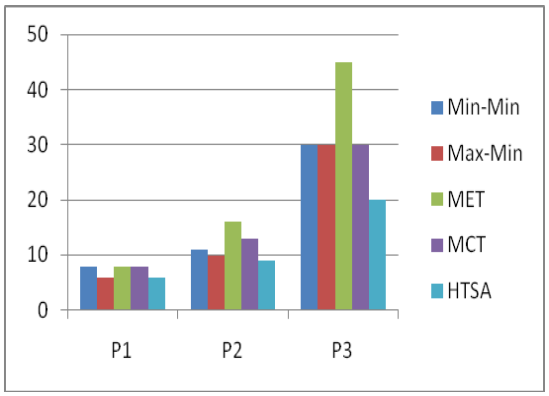

Figure 4. Comparative Analysis of heuristics (Makespan)

HTSA also balances load by using all the resources optimally. Resource utilization for all the problems is calculated and is shown in Table 3. It can be observed that HTSA uses all the available resources to the optimum and balances load among the resources. Figure 5 represent the resource utilization rate for all the problems.

Table 3. Resource Utilization in Percentage

Problem Set	Algorithm Used	R1	R2	R3
P1	Min-Min	100	0	
	Max-Min	100	66	
	MET	100	0	
	MCT	100	0	
	HTSA	83.3	100	
P2	Min-Min	54.54	100	
	Max-Min	100	80	
	MET	100	0	
	MCT	38.46	100	
	HTSA	100	100	
P3	Min-Min	66.67	0	100
	Max-Min	66.67	0	100
	MET	0	0	100
	MCT	66.67	0	100
	HTSA	100	100	75

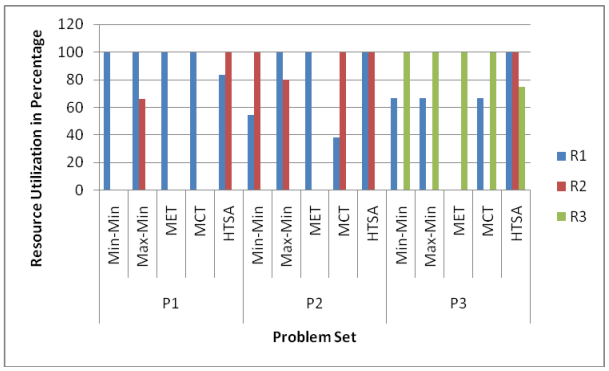

Figure 5. Analysis of Resource Utilization for problem sets

6. CONCLUSION AND FUTURE WORK

Task scheduling in grid computing environment is difficult for achieving high performance. Efficient task scheduling algorithm is needed to utilize the resource effectively and reduce overall completion time. In this paper, four scheduling algorithms are compared and a new scheduling algorithm that overcomes the drawbacks of the other entire algorithm is presented. This algorithm schedules the tasks with the aim of reducing makespan and balancing load. Experimental result shows that this algorithm minimizes makespan than other scheduling algorithms and uses resources effectively. Applying the proposed algorithm on actual grid environment and considering low and high machine heterogeneity can be the open problem in this area.

7. REFERENCES

[1] George Amalarethinam, D.I, Vaaheeda Kfatheen.S, "Max-Min Average Algorithm for Scheduling Tasks in Grid Computing Systems", International Journal of Computer Science and Information Technologies, Vol 3 (2), 2012, 3659-3663.

[2] T.Kokilavani, Dr.D.I.George Amalarethinam. "Load Balanced Min-Min Algorithm for static Meta – Task Scheduling in Grid Computing". International Journal of Computer Applications, Volume 20 – No.2. 2011.

[3] Geoffrey Falzon, Maozhen Li, "Enhancing list scheduling heuristics for dependent job scheduling in grid computing environments", Journal of Supercomputing, Springer, March 2010.

[4] Kamalam.G.K and Muralibhaskaran.V, , A New Heuristic Approach:Min-Mean Algorithm For Scheduling Meta-Tasks On Heterogenous Computing Systems, IJCSNS International Journal of Computer Science and Network Security, VOL.10 No.1, January 2010.

[5] Zhan Gao, Siwei Luo and Ding Ding, " A Scheduling Approach Considering Local Tasks in the Computational Grid" , International Journal of Multimedia and Ubiquitous Engineering, Vol 2, No. 4, October 2007.

[6] Siriluck Lorpunmanee, Mohd Noor Sap, Abdul Hanan Abdullah, and Chai Chompoo-inwai, "An Ant Colony Optimization for Dynamic Job Scheduling in Grid Environment", World Academy of Science, Engineering and Technology 29, pp. 314-321, 2007.

[7] 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.

[8] F.Dong and S.G.Akl, "Scheduling Algorithms for Grid Computing: State of the art and open problems", Technical Report of the Open Issues in Grid Scheduling Workshop, School of Computing, University Kingston, Ontario, January 2006.

[9] Dantong Yu and Thomas G. Robertazzi "Divisible Load Scheduling for Grid Computing", PDCS'2003, 15th Int'l Conf. Parallel and Distributed Computing and Systems. IASTED, pp.1 – 9, 2003.

[10] Braun, T.D., Siegel, H.J., Beck, N., Boloni, L.L., Maheswaran, M., Reuther, A.I., Robertson, J.P "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.

[11] M.Maheswaran, S. Ali, H.J.Siegel, D.Hensgen, and R.F.Freund, "Dynamic mapping of a class of independent tasks onto heterogeneous computing systems, J.Parallel Distribute Computing 59, 2 (Nov 1999), 107 – 121.

[12] R.F.Freund, and M.Gherrity, "Scheduling Resource in Multi-user Heterogeneous Computing Environment with Smartnet", In Proceedings of the 7th IEEE HCW 1998.

[13] Fahd Alharbi, "Simple Scheduling Algorithm with Load Balancing for Grid Computing" Asian Transactions on Computers (ATC ISSN: 2221-4275) Vol 02, Issue 2.