

Design and Implementation of an Efficient Resource Sharing algorithm for Grid Computing

Surendra Kumar Patel¹; Anurag Seetha²; Gupteshwar Gupta³

¹Department of Information Technology and Computer Application

Dr. C.V. Raman University, Kota Bilaspur, Chhattisgarh, India;

²Dept. of Computer Science and Engineering, Dr. C.V. Raman University, Kota Bilaspur, Chhattisgarh, India;

³Dept. of Mathematics, Govt. Tilda College, Raipur, Chhattisgarh, India

ABSTRACT

Grid Computing refers to a new technology infrastructure paradigm which is based on the Web and the Internet that coordinates and provides the facility of resource sharing over various geographical locations. Resource sharing and scheduling resources in Grid computing is a complex task due to the heterogeneous and dynamic nature of the resources. Resource sharing crisis brings Grid Technology that needs algorithms and mechanisms to be redesigned for resource handling. The analysis of algorithms is the determination of the number of resources (such as time and storage) necessary to execute them. In this paper, we can implement the MCT (minimum completion time) and MET (Minimum execution time) algorithm to increase the performance in terms of their speed of execution.

Keywords

Grid Computing, Resource Sharing, Load Balancing Scheduling, Gridsim

1. INTRODUCTION

Grid computing was introduced in the early 1990s by a supercomputing committee whose goal of using computing resources in a convenient form for calculations was complicated by the fact that the resources were distributed geographically [1].

The growth of internet and the availability of powerful computers and high speed network have made possible the use of geographically distributed and multi-owner resources to populate and solve large scale problems in science, engineering and commerce. The research was done on these topics which led to the emergence of a new paradigm known as Grid [2]. Grid computing technologies enable controlled resource sharing in distributed communities and the coordinated use of those shared resources as community members tackle common goals [3].

In order to fulfill the user expectations in terms of performance and efficiency, the Grid system needs efficient load balancing algorithms for the distribution of tasks. A load

balancing algorithm attempts to recover the response time of user's submitted applications by ensuring maximal utilization of available resources. The main objective is to prevent, if possible, the condition where some processors are overloaded with a set of tasks while others are lightly loaded or even idle. Although load balancing problem in conventional distributed systems has been intensively studied, new challenges in Grid computing still make it an interesting topic and many research projects are under way [4].

2. Load Balancing

Load balancing problem has been discussed in traditional distributed systems literature for more than two decades and various algorithms, strategies and policies have been proposed, classified and implemented. Load balancing algorithms can be classified into two categories, static and dynamic [5].

2.1 Static Load Balancing

Static load balancing algorithms allocate tasks of a parallel program to workstations based on either the load at the time nodes are allocated to some task, or based on average load of workstation cluster.

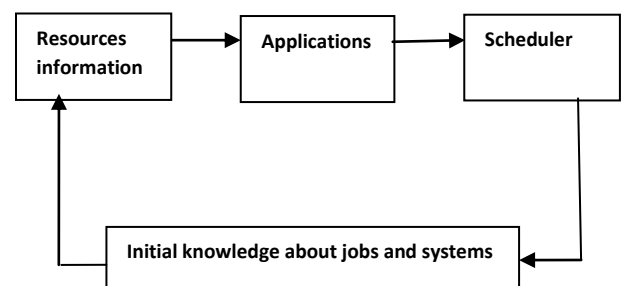


Fig. 1 Static Load Balancing

2.2 Dynamic load balancing

According to the name dynamic load balancing algorithms takes decision at run time, and use current or recent load information when making distribution decisions. In grid environment with dynamic load balancing allocate/reallocate resources at runtime based on no a priori task information, which determine when and which task has to be migrated.

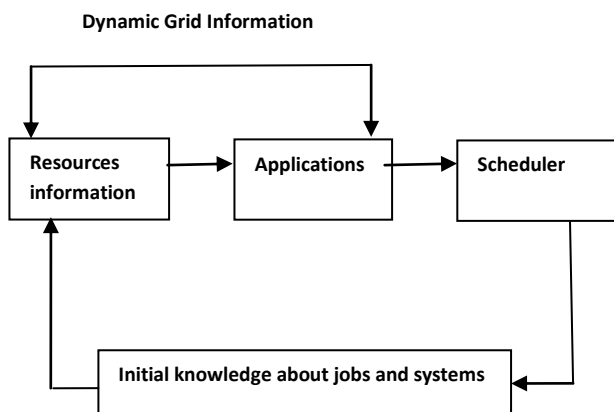


Fig. 2 Dynamic Load Balancing

3. Related work

A combined optimization algorithm using PSO and SA was proposed in Weijun et al. (2004) for the task scheduling problem. A particle consists of m segments and every segment has n different job numbers, representing the processing orders of n jobs on m machines. Thus, we have m machines and n jobs, which convert the continuous optimization problem to a discrete optimization problem [6]

Sahu et al. [7] introduced five major objective functions and Metrics to assess a grid scheduler performance, resource utilization, makespan and matching proximity looks usually have been used for performance measurement. Resource utilization is measured based on idle time of resources and effective job scheduling is measured by high resource throughput. Moreover, makespan is the time difference between the start time of the first job and the finish time of the last job [8], [9]. Matching proximity indicates the degree of proximity of a given schedule to the schedule produced by the Minimum Execution Time (MET) method [7].

To increase efficient usage of resources and mapping tasks different scheduling algorithms have been proposed. Among these heuristics, min-min [10], [11], [12], [13], [14], [15], [16] and Genetic algorithm [10], [11] achieve the best results in makespan and max-min for resource utilization [14].

The main objective of this study is to achieve efforts by using resource handling algorithms like MCT (Minimum Completion Time) and MET (Minimum Execution Time) that helps in achieving the goal of power savings and performance maximization in terms of their speed of execution.

4. Proposed methodology and Discussions

4.1 Design Methodology

Design Methodology diagram Fig. 3 represents the layered architecture of my research; this architecture consists of Jdk, Eclipse, Gridsim, operating system. Each and every layer is been inter linked. The first layer is the operating system on the top of this layer a GridSim is been placed this GridSim is nothing but the Grid Simulator this is used to perform any grid computational operations. The Java Development Kit (JDK) is an Oracle Corporation product aimed at Java developers. Jdk is developed by Sun Micro systems Jdk contains java virtual machine.

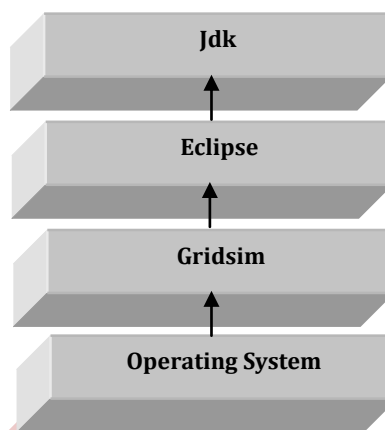


Fig. 3 System Design

In this research, a Grid resource refers to a processor and the phrase is used interchangeably. The terms job, task, and Gridlets are used interchangeably. The following assumptions are:

- Each job is non-pre-emptive, requiring one and only one machine at a time.
- Tasks have no priorities as linked
- Each machine can process at most one job at a time.
- The execution sequence of tasks on a processor is based on MCT (Minimum Completion Time).

In this research we have proposed two algorithms MCT (Minimum Completion Time) and MET (Minimum Execution Time) to reduce its waiting time and improved its execution and to improve Quality of Services (QoS) when system is perform Resource Sharing Management operation in Grid Model.

4.2 Parameters Used

- **Makespan:** the total running time of all jobs.

Let task set $T = t_1, t_2, t_3, \dots, t_n$ be the group of tasks submitted to scheduler and

Let Resource set $R = m_1, m_2, m_3, \dots, m_k$ be the set of resources available at the time of task arrival makespan produced by any algorithm for a schedule can be calculated as follows:

1) $Makespan = \max(CT(t_i, m_j))$

2) $CT_{ij} = R_j + ET_{ij}$

Where CT_{ij} is Completion Time of task i on resource j , ET_{ij} expected execution time of job i on resource j . and R_j is the ready time or availability time of resource j ; the time when machine m_j complete execution of all the prior assigned tasks.

- **Average waiting time:** the average waiting time spent by a job in the grid.
- **Success rate:** the percentage of jobs successfully completed in the system.
- **Grid resource and Grid user:**

Create a grid resource that encompasses of one or more machines. Each machine contains one or more processing elements. In this way, we have taken three machines. In Fig: 4 each machine contains three processing element. Each resource has number of processes, speed of processing and internal process scheduling policies. Grid user contains one or more Gridlets or jobs to be processed. Gridlet contains the job information or data. In this we can consider the three Gridlets Fig. 5.

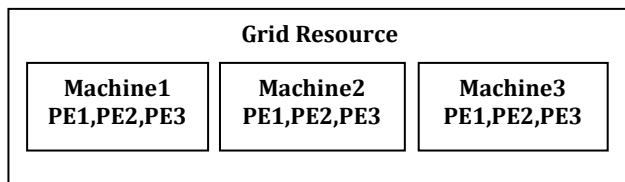


Fig. 4 Grid Resource

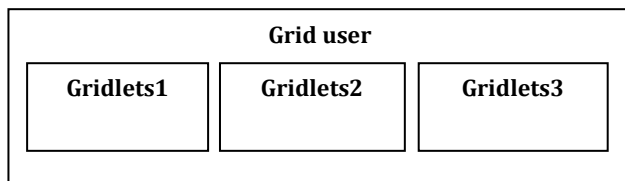


Fig. 5 Grid user

4.3 Grid scheduling algorithms

4.3.1 MET (Minimum Execution Time)

In this algorithm a task is assigned to the machine on which it can be executed in minimum time regardless of that machine's availability. Allocating job without considering machine availability might lead to load imbalance on grid machines [7], [10], [11].

4.3.2 MCT (Minimum Completion Time)

MCT assigns each task, in arbitrary order, to the machine with the minimum expected completion time (ready time of machine + job execution time on the selected machine) for that task. Allocating job in this manner may result in

execution of jobs on less fast grid machines [7], [10], [11].

5. Simulation Results and Discussion

5.1.1 Simulation Tool and Environment

Even though there are numerous tools available for simulating resource scheduling algorithms in Grid computing environments such as Bricks, OptorSim, SimGrid, GangSim, Arena, Alea, and GridSim, The simulation was carried out using the GridSim v4.0 simulator [17]. It provides facilities for modeling and simulating entities in grid computing environments such as heterogeneous resources, system users, applications, and resource load balancers which are used in designing and evaluating the task allocation algorithms. In order to appraise the performance of the proposed algorithm, a heterogeneous grid environment was built using different resource specifications. The resources differ in their operating systems, RAM, and CPU speed. In GridSim, tasks are modeled as Gridlet objects which contain all the information related to the task and the execution management details.

5.1.2 Performance evaluation and Analysis

Following are the results which describe the implemented algorithms in terms of Makespan that estimate the total running time of all jobs using MCT and MET algorithm.

- **Simulation results for Makespan of MCT algorithm with Grid environment for five numbers of jobs.**

Number of Gridlets	4	6	8	10	16
Makespan of MCT algorithm	11	8.8	7.2	6.5	5.5

- **Simulation results for Makespan of MET algorithm with grid environment for five Gridlets.**

Number of Gridlets	4	6	8	10	16
Makespan of MET algorithm	3.8	4.2	5.9	7.9	15

Experiments 1:

Figure 6 shows the result for makespan of MCT (Minimum Completion time) algorithm. This graph describes that makespan decreases by way of the increase in number of jobs in grid environment. In the below figure, x-axis represents the number of jobs and y-axis represents the total running time of all jobs in seconds.

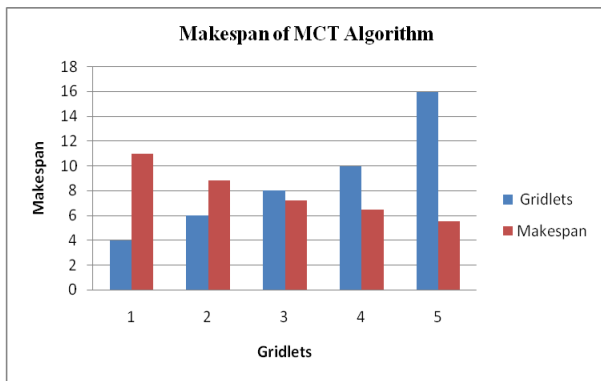


Fig. 6 Result for makespan using MCT algorithm

Experiments 2:

Figure 7 shows the result of Makespan of MET (Minimum Execution Time) algorithm. This result shows the completion time of five gridlets on grid environment which increases by way of the increase in the number of jobs. Makespan is the maximum time taken for the completion of all the tasks in a given application. In the below figure, x-axis represents the number of jobs and y-axis represents the total running time of all jobs in seconds

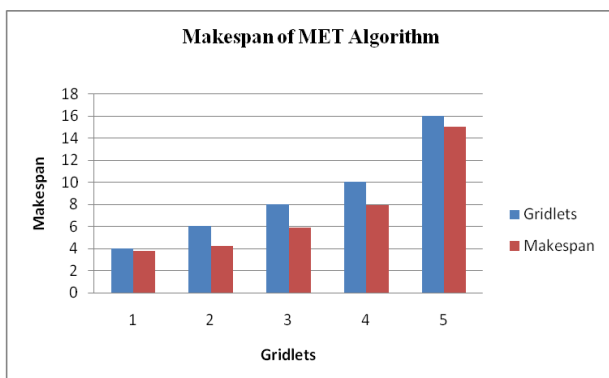


Fig. 7 Result for makespan using MET algorithm

6. Conclusion and future work

Grid Computing is the rapid growing technology, which shares the resources in the organization in an effective manner. Resource sharing requires more optimized algorithmic, otherwise the waiting time and response time are increased and the resource utilization is reduced, In order to avoid such reduction in the performances of the grid system, an optimal resource sharing algorithm is required. In this paper we implemented two algorithms MCT and MET algorithm to increase the performance in terms of their speed of execution. Based on the execution performance, us analysis both of the techniques and carried out results and found that makespan decreases when number of gridlet objects increase in MCT, while in MET makespan increases when number of jobs increases.

The future scope of this effort includes implementing the applications using various tools like Bricks, OptorSim, SimGrid, GangSim and Gridsim for showing the simulation,

Migration of jobs, Design and Testing of resource sharing in a Multi-middleware situation in Grid environment.

7. Acknowledgement

I am thankful to my guide Dr. Anurag Seetha and Dr. Guptaeshwar Gupta for their encouragement and valuable support.

8. References

- [1] Garg SK, Buyya R, Siegel HJ (2010) Time and cost trade-off management for scheduling parallel applications on utility Grids. *Future Gener Comput Syst* 26:1344–1355
- [2] Pinky Rosemarry, Payal Singhal, Ravinder Singh, Study of various job and Resource Scheduling Algorithm in Grid Computing, *International Journal of Computer Science and Information Technologies*, Vol. 3 (6), 2012,5504-5507 .
- [3] Ding, Yajun Li, Yuhang Yang Rongbo Zhu "A Hybrid Load balancing Strategy of Sequential Tasks for Computational Grids" 2009 International Conference on Networking and Digital Society.
- [4] B. Yagoubi and Y. Slimani "Task Load Balancing Strategy for Grid Computing" *Journal of Computer Science* 3 (3): 186-194, 2007 ISSN 1546-9239, 2007 Science Publications
- [5] Belabbas Yagoubi and Meriem Medebber "A Load Balancing Model for Grid Environment" 1-4244-1364-8/07/\$25.00 ©2007 IEEE
- [6] Weijun X, Zhiming W, Wei ZH, Genke Y (2004) A new hybrid optimization algorithm for the job-shop scheduling problem. In: *Proceeding of the 2004 American control conference*, vol 6, Boston, pp 5552–5557.
- [7] Sahu R., Chaturvedi A.K. ,Jan 2011, "Many-Objective Comparison of Twelve Grid Scheduling Heuristics ",*International Journal of Computer Applications* ", Vol113, pp. 9-17.
- [8] Pop F., Dobre C., Cristea V. , jun 2008, "Evaluation of Multi-Objective Decentralized Scheduling for Applications in Grid Environment", *IEEE, ICCP*, pp.231-238.
- [9] Chang H.J., Wu J.J., Liu P., Aug 2009, " Job Scheduling Techniques for Distributed Systems with Heterogeneous Processor Cardinality", 10th International Symposium on Pervasive Systems, Algorithms, and Networks , pp.57-62.
- [10] Braun T.D., Siegel H.J., Beck N., 2001, "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, pp.810-837.
- [11] Braun T., Siegel H., Beck N., Boloni L., Maheshwaran M., Reuther A., Robertson J., Theys M., Yao B., Hensgen D., R.Freund, 1999, "A Comparison Study of Static Mapping Heuristics for a Class of Meta-tasks on Heterogeneous Computing Systems", In 8th IEEE Heterogeneous Computing Workshop(HCW'99), pp. 15-29.

- [12] Izakian H., Ajith A., Vaclav S., 2009, " Comparison of Heuristics for Scheduling Independent Tasks on Heterogeneous Distributed Environments", International Workshop on HPC and Grid Applications(IWHGA2009), China, IEEE Press, pp. 8-12.
- [13] Izakian H., Ajith A., Vaclav S., 2009, "Performance Comparison of Six Efficient Pure Heuristics for Scheduling Meta-Tasks on Heterogeneous Distributed Environments" , Journal of Neural Network World, Volume 19, Issue 6, pp. 695-710.
- [14] Xhafa F., Barolli L., Durrezi A., 2007, "Batch Mode Scheduling In Grid Systems", International Journal Of Web & Grid Services, Volume 3, No 1, pp19-37.
- [15] Maheswaran M., Ali S., Siegel H.J., Hensgen D., Freund R.F., 1999, "Dynamic mapping of a class of independent tasks onto heterogeneous computing systems", Journal of Parallel and Distributed Computing 59,Volume 9, Issue 3, pp 107-131.
- [16] Meihong W., Wenhua Z. , 2010, "A comparison of four popular heuristics for task scheduling problem in computational grid ", 6th International Conference on Wireless Communications Networking and Mobile Computing (WiCOM) ,IEEE, pp 1-4.
- [17] R. Buyya, "A grid simulation toolkit for resource modelling and application scheduling for parallel and distributed computing", www.buyya.com/gridsim/, accessed on January '2014.