



Optimized Performance Task Scheduling algorithm for Grid Computing

Kulvinder Singh and Sonia,
Department of Computer Science & Engineering
U.I.E.T Kurukshetra University, Kurukshetra Haryana, India

Abstract- Grid computing is the organization of computer resources from multiple locations to reach a common goal. The grid is like a distributed system with non-interactive workloads that involve a large number of files. Grid computing is different from conventional high performance cluster computing systems such as is that grids tend to be more loosely coupled, heterogeneous, and geographically distributed. Grid is multiple numbers of computers having same class and is clustered together. A grid computer is connected through a super-fast network and shares the devices like disk drives, mass storage, printers and RAM. Grid Computing is more cost efficient with respect to Super Computing [1]. Scheduling algorithms are mostly preferred for scheduling the tasks of a parallel program onto the heterogeneous computing systems as they are relatively simple to use, these provide good quality of schedules in less time complexity than the other approaches. In This Paper we discuss the Optimized Performance Task Scheduling. OPTS is an algorithm that will do optimized prioritization means priorities of the tasks will be decided so as to achieve optimized grid scheduling. In OPTS algorithm, priorities to the tasks of a schedule will be decided and also we will take several parameters to decide that, which task will be allocated to which processor from the available processors for that particular schedule. Optimized Performance Task Scheduling algorithm will be designed with three phase's namely level sorter phase, task prioritization phase and processor selector phase. This algorithm identifies the tasks to be executed in parallel using the level sorter phase. Then it will prioritize the tasks in the task graph in the prioritization phase. Processor selector phase selects the tasks in the order of their priorities and schedule each selected task onto the best scheduled processor for execution. This algorithm reduces the time complexity by achieving optimized results. Reliability factor is also considered in this algorithm that will be used during processor selection for a task.

Keywords: Grid Computing, Task Scheduling, List Scheduling Algorithms, HPS, PETS, OPTS, DAG.

I. Introduction

A computational grid is a hardware & software infrastructure that provides a faithful, consistent, persistent, and inexpensive access to high-end computational competencies." [2]. Many research projects require lot of CPU time and a lot of memory and some projects need the ability to communicate in real time. The super computers used today are not enough to solve those needs. Super Computers don't have the capacity, however if they, it would not be economical justifiable to use these resources.

Computational grids are the solution to all these problems and many more. They offer a convenient way to connect many devices (e.g., processors, memory and IO-devices) so that end users is able, if allowed, to use all the devices computational powers combined for a certain amount of time. For example if a researcher needs to make some very CPU consuming calculations, he could occasionally borrow CPU-time from the grid to a much lower cost than borrow the time from a super computer. A grid could be created in all environments where end users have a computer with memory and CPU. Computers are often in idle state and over time they only use 5% of their capacity. Hence there is a slice of computational power successful to waste, set up as a computational net each end user would be able to enjoy a lot of computational power that otherwise would have gone to waste.

Grid computing is the federation of computer resources from multiple sites to reach a common goal. The grid is like a distributed system with non-interactive workloads that involve a large number of files. Grid computing is different from conventional high performance cluster computing systems such as is that grids tend to be more loosely tied, assorted and distributed. Although a single grid can be dedicated to an individual application, commonly a grid is

used for a variety of determinations. Grids are often constructed with general-purpose grid middleware software libraries. Computational Grid appeared to satisfy the increasing demand for bandwidth, storage, and computational assets.

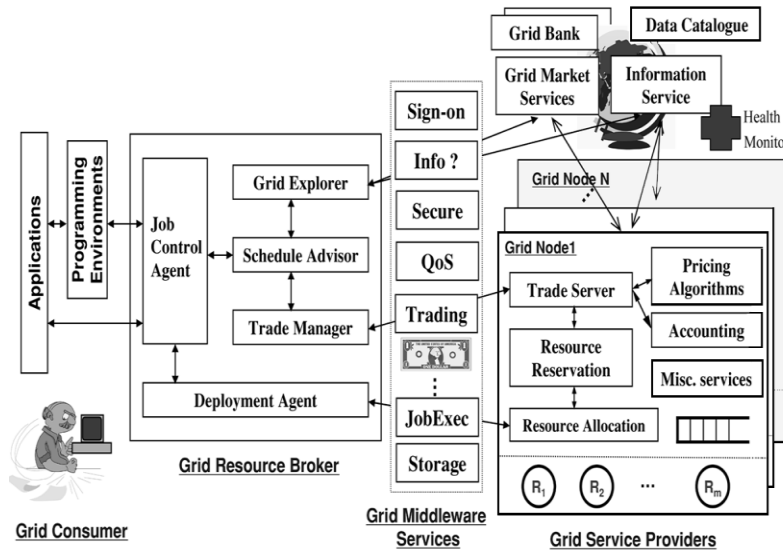


Fig 1: Grid Architecture [2]

II. Scheduling Algorithm

A. Task Scheduling

A task partitioning algorithm can be hired to partition a parallel application into a set of precedence-constrained tasks represented in the form of a Directed Acyclic Graph(DAG), while a scheduling algorithm can be used to schedule the DAG onto the processors of the DHCS in order to minimize the make span or schedule length of an application. Make span is defined as the time at which all tasks have finished their execution. The process of scheduling a DAG onto a distributed system .Here the vertices in the DAG represent the tasks and the edges represent the amount of data (output) to be transferred from one task to another. Each task is labeled with a task number and the estimated computation time for its execution. The target system consists of four completely connected processors and for drawing it is assumed that each task takes the same amount of execution time on all the four Processors [4, 5].

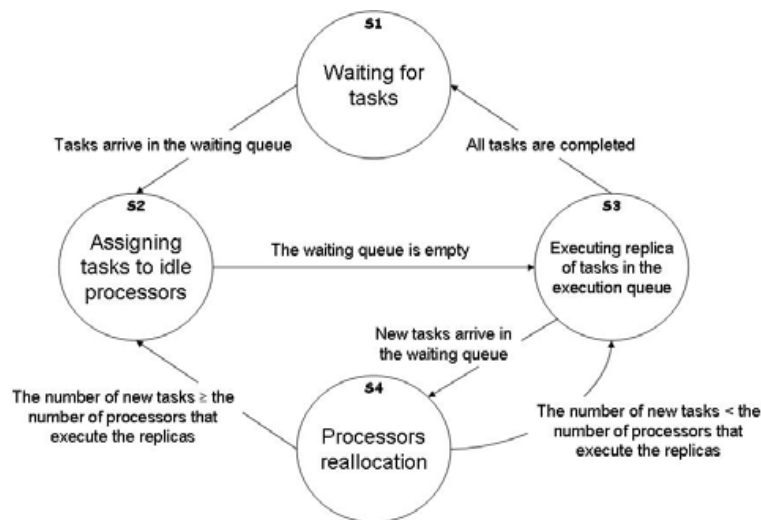


Figure 2: Task Scheduling [6]

B. List Scheduling Algorithms:

One of the earliest proposed solutions to the task scheduling problem is the list-scheduling algorithms. Interestingly, a majority of task scheduling algorithms developed for the homogeneous and the heterogeneous processors belong to the class of list-scheduling algorithms. This is probably due to their relative simplicity and low complexity in comparison with other approaches. The basic idea in the list-scheduling is to assign priorities to the tasks of the DAG and place the tasks in a list arranged in decreasing order of priorities. At each scheduling step the ready task (a ready task is one whose predecessor tasks have finished the execution and the communications data are available at the processor to which the task is to be assigned) with the highest priority is selected and scheduled to a suitable processor. If two or more tasks have equal priority, then the tie is resolved by selecting the task randomly or selecting the task that takes more execution time. The process is repeated until all the tasks are scheduled to the suitable processors. There are two important Issues in the list-scheduling algorithm: (1) How to prioritize the tasks of the DAG? (2) How to select the best processor for executing the selected task?

Earliest Time First (ETF) algorithm computes the earliest start times, at each scheduling step, for all the ready tasks and then selects a task with the smallest earliest start time. When two or more tasks have the same value of their earliest start times, the ETF algorithm breaks the tie by scheduling the task with the highest static level. While selecting the best processor in heterogeneous system a commonly used mechanism is EFT.

C. Highest Level First with Estimated Times (HLFET) Algorithm:

HLFET is one of the earliest and simplest scheduling algorithms developed for the fully connected homogeneous processors. In the HLFET the priority is assigned to the task based on the sum of Computation costs of all the tasks along the longest path from the task to an exit task. The algorithm schedules a task to a suitable processor that allows the earliest start time. The main drawback of the HLFET algorithm is that it ignores the Communication costs on the edges.

D. High Performance task Scheduling (HPS) Algorithm:

The HPS algorithm is an application scheduling algorithm proposed for the DHCS consisting of bounded number of completely connected processors. The goal of the algorithm is to provide better schedule length with lesser time complexity. The HPS algorithm is designed with three phases namely: Level Sorter phase, Task Prioritization Phase, Processor Selection Phase. This algorithm firstly identified the tasks to be executed in parallel and then it had decided priorities and finally it had decided that which task will be allocated to which processor. The task prioritization phase prioritizes the task using the attributes such as DLC (Down Link Cost), ULC (Up Link Cost) and the LC (Link Cost).

E. Performance Effective Task Scheduling (PETS) Algorithm:

PETS is similar to the HPS algorithm and consist of three phases: Level Sorter phase, Task Prioritization Phase, Processor Selection Phase. The Level Sorter Phase and Task Prioritization phases were similar to HPS algorithm; Processor Selection Phase of PETS is different from HPS. The PETS algorithm uses the priority of the parent tasks (immediate predecessor tasks) and the data transfer cost to the successors tasks (immediate child tasks) while assigning priority to the tasks in addition to other parameters.[7].

Estellon *et al.* [8] explained a problem of task scheduling in which human resource allocation is addressed. The problem that is measured is very hard: it consists of several NP-hard sub-problems. An effective and efficient local-search experiential approach is described to solve this problem. An algorithm is used here that explains the solution for the considered problem. This algorithm reduces the configuration time, data dependency between tasks, inter-task communication and task resource utilization. The algorithm uses the dynamic programming method.

Huang (2010) [9] proposed work on task scheduling. This paper explained the Reduced Data Movement Scheduling (RDMS) algorithm. When the hardware tasks of an application cannot concurrently fit in an FPGA, the task graph needs to be partitioned and scheduled into multiple FPGA configurations, in a way that lessens the total execution time. This paper described the Reduced Data Movement Scheduling (RDMS) algorithm that aims to improve the overall performance of hardware jobs by taking into account the reconfiguration time, data dependency between tasks, inter-task communication as fine as task resource utilization.

III. Proposed Work

Grid computing is the organization of computer resources from numerous locations to reach a common goal. The grid is similar to a distributed system with non-interactive workloads that involve a large number of files. Grid computing is different from conventional high performance cluster computing structures such that grids tend to be

more loosely attached, heterogeneous, and geographically distributed. Task scheduling is the key issue in obtaining high performance in heterogeneous systems. Task scheduling in heterogeneous systems is a NP-problem. The efficiency of accomplishing parallel applications on heterogeneous systems critically depends on the methods used to schedule the jobs of a parallel application. The objective is to lessen the overall completion time or make span.

Scheduling algorithms are usually preferred for scheduling the tasks of a parallel program onto the heterogeneous computing systems as they are relatively simple to understand and to use; these provide good quality of programs in less time complexity than the other approaches. Static DAG scheduling has focused on finding suboptimal solutions to obtain a good solution in an adequately short time. List scheduling heuristics usually generate good quality schedules at a reasonable cost. In comparison with clustering algorithms, these have lower time complexity and in contrast to task duplication strategies, their solutions use fewer processors, producing more efficient schedules.

Our proposed work consists of building an algorithm that will assign the tasks to the processors available, in order to obtain optimized performance in terms of time, cost and efficiency. Optimized schedule is the requirement of every scheme. Researchers have developed many list scheduling algorithms. Three main facts during designing of Optimized Performance Task Scheduling algorithm in grid computing will be considered:

- (1) Allocate priorities to the tasks of a schedule.
- (2) Select the task on the foundation of priority.
- (3) Which task will be allocated which CPU?

We will minimize the cost function. If priority of two tasks will be same, then we will resolve it by selecting one out of them randomly. Optimized Performance Task Scheduling is an algorithm that will do optimized prioritization means priorities of the tasks are decided so as to achieve optimized grid scheduling. In OPTS algorithm, priorities to the tasks of a schedule will be decided and also we will take several parameters to decide that which task will be allocated to which processor from the available processors for that particular schedule. Optimized Performance Task Scheduling algorithm will be designed with three phases.

References:

- [1] en.wikipedia.org
- [2] Foster, I. (Mathematics and Computer Science Division Argonne National Laboratory Argonne) and Carl Kesselman (Information Sciences Institute University of Southern California Marina Del Rey), "Computational Grids", pp. 1- 29, 1998.
- [3] Buyya, Kumar R., Abramson D. and Venugopal S., "The Grid Economy" on the proceedings of IEEE, VOL. 93, NO. 3, pp. 698- 714, 2005.
- [4] Graham, R., Rinnooy Kan, A., Lawler E. and Lenstra, J. K., "Optimization and approximation in deterministic sequencing and Scheduling: A Survey", *Annals of Discrete Mathematics*, vol. 5, pp. 287-326, 1979.
- [5] Guy, M. R. and Johnson, D. S., *Computers and Intractability: "A guide to the Theory of NP-Completeness "*, W. H. Freeman and Co., 1979.
- [6] Bansal, S., Kothari, B. and Hota, C. , "Dynamic Task-Scheduling in Grid Computing using Prioritized Round Robin Algorithm ", vol. 8, Issue 2, March ,2011.
- [7] E. Ilavarasan, "Task Scheduling Algorithms For Distributed Heterogeneous Computing Systems", A Thesis, pp. 1-181, September, 2007.
- [8] Estellon, B., Gardi, F. and Nouioua, K., "High-performance local search for task scheduling with human resource allocation" ,pp. 1-15,2009.
- [9] Huang, M., Narayana, K. V., Simmler, H., EL-Ghazawi, S. & K., "Reconfiguration and Communication-Aware Task Scheduling for High-Performance Reconfigurable Computing ",vol. 3,issue 4,pp. 1-24,2010.
- [10] www.google.co.in