



Improvement to Group job scheduling in Grid computing

D. L. Veer¹, Prof. H.A. Akarte²

Department of Computer Engineering,

Dr. Babasaheb Ambedkar Technological University

Lonere, Raigad (Maharashtra),

INDIA

Abstract: Grid computing is becoming the most important research area in the high performance computing. Under his concept, job scheduling in Grid computing has more complicated problems to discover a diversity of available resources, select the appropriate applications and map to suitable resources. However the major problems are the optimal job scheduling in which Grid nodes need to allocate the appropriate resources for the job group.

In this paper, we have proposed a scheduling algorithm to maximize the resource utilization and minimize the processing time of jobs. This algorithm is based upon the grouping of jobs. These grouping are mainly based upon network, processing time and memory size.

Keywords: grid computing, job scheduling, adaptive scheduling, Resource utilization.

I. Introduction

The last decade has seen a substantial increase in commodity computer and network performance, mainly as a result of faster hardware and more sophisticated software. Nevertheless, there are still problems in the fields of science, engineering and business, which can't be effectively dealt with using the current generation of supercomputer. In fact, due to their size and complexity, these problems are often resource (computational and data) intensive and consequently entail the use of a variety of heterogeneous resources that are not available in a single organization. The ubiquity of the Internet as well as the availability of powerful computers and high-speed network technologies as low-cost commodity components is rapidly changing the computing landscape and society.

These technology opportunities have led to the possibility of using wide area distributed computers for solving large-scale problems, leading to what is popularly known as Grid computing. Grids enable the sharing, selection, and aggregation of a wide variety of resources including supercomputers, storage systems, data sources, and specialized devices that are geographically distributed and owned by different organizations for solving large-scale computational and data intensive problems in science, engineering, and commerce. Grid is a type of parallel and distributed system that enables the sharing, selection, and aggregation of geographically distributed "autonomous" resources dynamically at runtime depending on their availability, capability, performance, cost and user's quality-of-service requirements. Grid applications are special class of distributed applications that has high computing and resource requirements, and are often collaborative in nature. Networks connect resources on the Grid, the most prevalent of which are computers with their associated data storage. Although the computational resources can be of any level of power and capability, some of the most interesting Grids for scientists involve nodes that are themselves high performance parallel machines. These provide major resources for simulation, analysis, data mining and other compute-intensive activities. Grid Computing enables devices, regardless of their operating characteristics to be virtually shared, managed and accessed across an enterprise, industry or workgroup.

This virtualization of resources places all of the necessary access, data and processing power at the groups of those who need to rapidly solve complex business problems, conduct compute-intensive research and data analysis, and engage in real-time. The grid computing enables the virtualization of distributed computing resources such as processing, network bandwidth and storage capacity to create a single system image, granting users and applications seamless access to vast IT capabilities. Just as an Internet user views a united instance of content via the Web, a Grid user essentially sees a single, large virtual computer.

1.1 Types of Grid

- Computational grid - In which machines with set-aside resources stand by to crunch data or provide coverage for other intensive workloads.
- Scavenging grid - Commonly used to scavenge CPU cycles from idle servers and desktop machines for use in resource-intensive tasks.
- Data grid - Which provide a unified interface for all data repositories in an organization, and through which data can be queried, managed, and secured.

II. Related Work

In the field of grid computing there are many algorithms have been proposed. Each proposed algorithms have various capabilities and restrictions. Jobs submitted to a grid computing system need to be processed by the available resources. Best resources in terms of processing speed, memory and availability status are more likely to be selected for the submitted jobs during the scheduling process. Best resources are categorized as optimal resources.

A scheduling optimization method should consider the following two aspects, one is the application characteristics, and the other is the resource characteristics [1].

A dynamic job grouping-based scheduling algorithm groups the jobs according to MIPS (million instructions per second) of the available resources. This model reduces the processing and communication time of the job, but this algorithm doesn't take the dynamic resource characteristics into account and the grouping strategy may not utilize resource sufficiently [2].

A Bandwidth-Aware Job Grouping Based scheduling strategy schedules the jobs according to the MIPS and bandwidth of the selected resource, and sends job group to the resource whose network bandwidth has highest communication or transmission rate. But, the strategy does not ensure that the resource having a sufficient bandwidth [3].

2.1 Types of scheduling

Grid computing scheduling is of three types centralised, distributed and hierarchical scheduling [9].

In central scheduling central machine controls the scheduling jobs and act like resource manager. Jobs are entered into central machine and allocated to resources according to scheduling policy at central node.

Distributed scheduling there is central scheduler and resource manager. There are many local schedulers which are interacting for each other for scheduling and resource information.

Hierarchical scheduling is mixture of centralized and distributed scheduling. In this scheduling local schedulers contact to meta-scheduler so it can send jobs to local schedulers [9].

The above analysis of various grouping based job scheduling strategy presents some of their advantages and disadvantages. Some of the algorithms don't take the dynamic resource characteristics into account, some of the grouping strategy can't utilize resource sufficiently and some of the algorithms do not pay attention to the network bandwidth and memory size. To solve the above problems, an adaptive job scheduling is proposed.

2.2 Stages of Scheduling

- **Resource Discovery**
Resource discovery involves the user selecting a set of resources to investigate in more detail in phase two, information gathering. At the beginning of this phase, the potential set of resources is the empty set, and at the end of this phase, the potential set of resources is some set that passed a minimal feasibility requirement.
- **System Selection**
Given a group of possible resources (or a group of possible resource sets), all of which meet the minimum requirements for the job, a single resource (or single resource set) must be selected on which to schedule the job. This is generally done in two steps: gathering information and making a decision.
- **Run Job**
The third phase of scheduling is running a job. This involves a number of steps, few of which have been defined in a uniform way between resources. They include: Making an advance reservation (optional), Submitting job to the resources, Preparing tasks, monitoring progress, and Completion of tasks.

2.3 Problem

Scheduling algorithms are responsible for mapping the jobs to the resources. In the existing scheduling algorithms, the nature of job is not considered while allocating them to resources. Hence if a heavy job is allocated to a resource with low processing capability then it will decrease the overall efficiency of the scheduling.

The above analysis of various grouping based job scheduling strategy presents some of their advantages and disadvantages. Some of the algorithms don't take the dynamic resource characteristics into account, some of the grouping strategy can't utilize resource sufficiently and some of the algorithms do not pay attention to the network bandwidth and memory size. To solve the above problems, an adaptive job scheduling is proposed.

III. Job Scheduling Mechanism

User creates a list of jobs in the user machine, these jobs are sent to the job scheduler for scheduling arrangement. The job scheduler obtains information about the available resources from the Grid Information Service (GIS).

The grouping characteristics will be based on the resource and network characteristics. In case of resource characteristics processing power and available memory size and in case of network characteristics bandwidth (Mb/sec) will be considered. After selecting the above information jobs are grouped. Job groups can be formed until

1. The processing power of resources should be less than resources capacity.
2. Processing time of job groups should be less than expected time.
3. Job group size will not grow than the available resource memory size.

IV. Proposed Algorithm

Grouping of the job is done on processing speed, network bandwidth and memory size available at the resource. After gathering the total information about resources and jobs to be done. System organizes the jobs in the first come first served policy to form groups. Resources may be organized in their decreasing order of processing speed that is fastest resources are allocated firstly.

The actual grouping of job done in condition that job groups are formed up to the processing power of the resources will be greater than or equal to jobs in the group.

The grouped jobs before allocating to the resources must satisfy the conditions

$$\begin{array}{ll} \text{proc_req_jobgroup} \leq \text{proc_reso} * \text{time} & \dots\dots\dots 1 \\ \text{job_group_size} = \text{network_speed} * \text{time} & \dots\dots\dots 2 \\ \text{memory_job_group} \leq \text{memory_resources} & \dots\dots\dots 3 \end{array}$$

In the first condition states that the processing speed of the resources must be greater than or equal to processing time required for completion of the job group so that required job group doesn't exceed the maximum time. similarly in the second condition says that the size of the job group at the scheduler before transfer is same as the size at the resource and last condition states that memory available at resource is more than or equal to the memory required for job execution.

In this algorithm initially scheduler receives the information about available resources by grid information service (GIS) and sorts the job in descending order of their number of instructions to be executed. After getting the information of resources in the next step these jobs are grouped based upon the first come first served policy (FCFS). Jobs inserted into group up to when these can satisfy the condition of resource utilization is less or equal to capacity of the job group.

In this way jobs are inserted into group until these job groups can satisfy the condition of the memory size, processing speed and bandwidth of the resources .after forming these groups scheduler gives this groups to resources. After the job completion the results are collected by scheduler and send to corresponding users. Now these resources are available for grid computing.

Algorithm

1. Jobs are submitted to scheduler.
2. Scheduler gets the information about the resource status.
3. Jobs are sorted in descending order according their number of instructions.
4. Groups of jobs to be formed based upon the condition of processing power, bandwidth and memory availability at resource.
5. Send these jobs to the particular resource.
6. Receive computed job group from resource
7. Repeat steps 4,5,6.

Resource manager keeps the information about new resource and informs the scheduler to form job group to assign to this new entered resource. If some large job doesn't satisfy the above conditions then are rotated until they get some suitable resource in grid.

V. Conclusion

To utilize grid resources efficiently, an adaptive job grouping scheduling algorithm is proposed. The proposed Scheduling algorithm in Grid Computing is a grouping based job scheduling strategy considers memory constraint of individual jobs together with expected execution time at the job level rather than at the group level and network constraints. The grouping algorithm improves the processing of fine grained jobs.

VI. References

- [1] N. Muthuvelu, Junyan Liu, N.L.Soe, S.venugopal, A.Sulistio, and R.Buyya "A Dynamic job grouping-based scheduling For deploying applications with fine grained tasks on global grids,".
- [2] Ng Wai Keat, Ang Tan Fong, "Scheduling Framework For Bandwidth- Aware Job Grouping-Based Scheduling In Grid Computing".

- [3] D. Nithyapriya , N.Krishnamoorthy “A dynamic job grouping-based scheduling for deploying applications with fine grained tasks on global grids,”.
- [4] Srikummar Venugopal, “Scheduling Distributed Data-Intensive Applications on Global Grid,”
- [5] Hojjat Baghban, Amir Masoud Rahmani “A Heuristic on Job Scheduling in Grid Computing Environment”.
- [6] Jing Wang, Gongqing Wu, Bin Zhang, Xuegang Hu “A heuristic algorithm for scheduling on grid computing environment”.
- [7] X. L. Shi and Y. Zhao, “Evolutionary Grid Scheduling Algorithm with Predictive Resource Optimization”
- [8] Zhen Dong, Chenyang Zhao “A Fully Decentralized Scheduling Approach for Grid Computing”
- [9] Maozhen. Li, Mark. Baker, “The Grid”.
- [10] Manoj Kumar Mishrai, Raksha Sharmai, Vishnu Kant Son “A Memory-Aware Dynamic Job Scheduling Model in Grid Computing”.
- [11] Angelos Michalas, Malamati Louta
- [12] “Adaptive Task Scheduling in Grid Computing Environments”.
- [13] Dan Garlasu , Virginia Sandulescu, Ionela Halcu, Giorgian Neculoiu, Oana Grigoriu “A Big Data implementation based on Grid Computing”.