Optimized Processor scheduling for job management in Grid computing

Daljit Singh[1] , Sarpreet Singh[2]

[1]Research Scholar, Dept. of Computer Science & Engineering, Shri Guru Granth Sahib World University, Fatehgarh Sahib.
[2]A.P Dept. of Computer Science & Engineering, Shri Guru Granth Sahib World University, Fatehgarh Sahib.

ABSTRACT: Grid computing is an extension of distributed computing where every computer on the network shares all its resources turning the network more powerful. A network can be hardwired or wireless (over the internet). System can also be homogenous or heterogeneous. Scheduling in Grid computing is challenging task and include many scheduling types such as ask scheduling, cpu scheduling, job scheduling etc. Major issue with the grid environment is the effective utilization of the scheduling of central processor units available. The problem can be addressed by scheduling the jobs to those nodes and processors that are lightly loaded. In Job processing management, selection of efficient processor or sub processor is a challenging part. In our research we will introduce optimized priority based cpu scheduling scheme which will be based on the selection of processors. For better communication and availability of processor, we will manage the cpu scheduling algorithms (i.e. SJF (shortest job first), PRIORITY, and FCFS (first come first serve)) for given data set according to the priority of job assigned. For avoiding lack of processor resources, for better utilization of processor resources, we have used a load balancing approach which will compare the utilization of processors. If utilization of resources in central processing units is more than its load handling capacity then we have shuffled the processor according to the job priority.

Keywords: *Grid Computing, Shortest Job First Scheduling, First Come First Serve Scheduling, Job Management, Priority based Scheduling.*

1. INTRODUCTION

A grid is a collection of nodes connected via a network and managed by a resource broker. It promotes the sharing of services, computing power and resources such as disk storage databases and software applications. A node is the most basic component in grid computing. It is a collection of work units that can be shared and that can provide some capabilities. As Grid computing is an extension of distributed computing where every computer on the network shares all its resources turning the network more powerful. A network can be hardwired or wireless (over the internet). System can also be homogenous or heterogeneous. If two grid computing systems do not follow same set of protocols then they may not be compatible with each other. Many organizations are working together in the same direction to create standard protocols that make it easier to set up grid environments. In grid computing, at least one computer acts as a server which allocate the resources to all jobs that are ready to execute and according to job request number of resources will be provided to it by the available processor.

In related study, a grid scheduling architecture has proposed. User first login with a genuine username and password and after successful authentication, the job is collected by job collector. The Job collector maintains all the resource requirement of the jobs. Then global scheduler schedules the job according with the algorithm used like FCFS, resources. When the job is finished, the user is informed. Grid Resource Database maintains all the basic

details about the nodes like CPU frequency, SJF, etc. Job of resource discovery is to find suitable nodes for the execution of job that is ready to execute. Once the resource list has been assembled, the optimal resources that meet the user's requirements such as cost are selected from it. Once the job and resources have been selected, input job is transferred to and run on the number of sub processors, RAM, IP Address and Available CPU Resource (ACR). ACR = (frequency of CPU * CPU idle time)/100.

In particular, we will manage the cpu scheduling algorithms (i.e. SJF (shortest job first), PRIORITY, and FCFS (first come first serve)) for given data set according to the priority of job assigned. For avoiding lack of processor resources, we will use a threshold utilization value which will be compared with utilization of cpu. If utilization of resources in cpu is more than threshold then we will shuffle the processor according to the job priority.

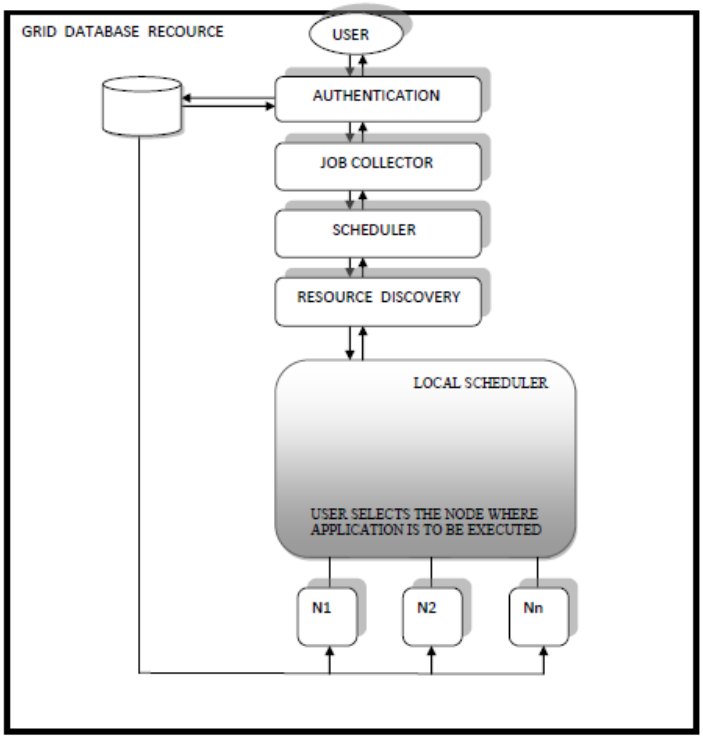

Fig. 1 Grid architecture with scheduling [1]

Some of the scheduling techniques are explained below:

- **First Come First Serve (FCFS)** or also known as First In First Out (FIFO) is the simplest and the most fundamental of grid scheduling that involves client-server interaction [7]. In grid scheduling, FCFS policy manages the jobs based on their arrival time, which means that the first job will be processed first without other biases or preferences. This concept has been used by several well known enterprise scheduler such as MAUI and PBS [8].
- **Shortest job First (SJF)** also known as Shortest Job Next (SJN) or Shortest Process Next (SPN) is a scheduling technique that selects the job with the smallest execution time [4]. The jobs are queued with the smallest execution time placed first and the job with the longest execution time placed last and given the lowest priority [5]. In theory, the best strategy to minimize the response time is to schedule the shortest job on the fastest resource [9]. Since this policy gives preference to some groups of jobs over other group of jobs, this policy is unfair when compared to FCFS policy [10]. In extreme cases, when jobs with shorter execution time continue to arrive, jobs with longer execution period may never get a chance to execute and would have to wait indefinitely. This is known as „starvation" and would pose a serious problem and reflect the low degree of fairness for this policy [7]. In addition, SJF is believed to have the maximum makespan time compared to other algorithms because of this characteristic.

2. PROBLEM DEFINITION

The grid is an emerging technology for enabling resource sharing and coordinated problem solving in dynamic multi-institutional virtual organizations. The resource-matching problem in the grid involves assigning resources to tasks in

order to satisfy task requirements and resource policies. Scheduling system plays a vital role to allocate resources to the input jobs. The goal of scheduling is to achieve maximum throughput with the available computing resources. One major issue with the grid environment is the effective utilization of the scheduling of central processor units available. The problem can be addressed by scheduling the jobs to those nodes and processors that are lightly loaded. In Job processing management, selection of efficient processor or sub processor is a challenging part. In our research we will introduce optimized priority based cpu scheduling scheme which will be based on the selection of processors. For better communication and availability of processor, we will manage the cpu scheduling algorithms (i.e. SJF (shortest job first), PRIORITY, and FCFS (first come first serve)) for given data set according to the priority of job assigned. For better utilization of processor resources, we have used a load balancing approach which compared the utilization of cpu. If utilization of resources in cpu is more than its load handling capacity, which will affect the cpu performance then we have shuffled the processor according to the job priority. Further we have compared our proposed techniques with Min-Min and Greedy scheduling algorithms.

3. SIGNIFICANCE AND OBJECTIVES

This research is very useful in grid computing processor utilization for higher resources utilization in better way. Many scheduling policies tends to schedule the processor according to the utilization but lacks in providing priority scheduling which leads to the performance decrease. The main focused objectives of the research are to find the better scheduling for Processor utilization based on the job priority defined by system and to find the solution for better scheme based on processor migration based on low utilization of CPU.

4. RESULTS AND DISCUSSION

Our research starts with important information fetching for scheduling of the processors. We have used cloudsim simulator environment for building required grid environment. In our program we have proposed a scheduling technique that will evenly distribute the load over all the resources, by determining a schedule for currently available tasks. We are proposing a method that will create a balance between CPU load and make-span.

A schedule is created to find out how we should run tasks over the machines, so that our tasks may finish with in the minimum possible time. It is quite obvious that running the task on best configuration machines will reduce the task's make-span. But sending maximum number of tasks over a single machine may result in machine failure or decreased machine performance. So instead of sending all the tasks over few machines if tasks will be distributed over all the machine in some well planned way results will be better.

Scheduling of the tasks over the resources can be considered a two step process. In first step we find out which task has to be scheduled and in second step we find out a resource for that task.

In our program for task selection we have used some well known techniques that are FCFS, Shortest Job First, and a priority based technique. By using these techniques we pick a resource that has to be scheduled in the current iteration.

In the second step that is the main part of the program, we are finding the resource that will run the task within minimum make-span. According to our approach we will send tasks to the best machine till it can handle the load in best possible way, as running tasks over best machine will execute task with minimum make-span. At some point of time when load on the best resource will exceed an acceptable limit, it's performance will decrease that will be considered as a threshold limit and task will be assigned to another machine that will give better result than the current selected resource.

In each iteration we are trying to find out a resource that will finish our task within minimum make-span this way we will be able to finish all jobs within minimum make-span. Moreover as the performance of the current resource will decrease that will signal excessive load over the machine and at that time we will find some other resource that can execute our tasks in better way this will keep load under acceptable limit.

In this way our program will keep a balance between load and make-span. Further we have used Min-Min and Greedy approach to compare our program. And most of the time our make-span is better than these approaches.

Simulation Run	Scheduling using SJF	Scheduling using FCFS	Scheduling using Priority	Scheduling using Greedy	Scheduling using Min-Min
1st	1320	1281.14	1043.	20166.62	3000

Table 1: Results of Scheduling with various techniques

We have run the simulation 10 times for each algorithm and these results are average make-span. First three are our proposed algorithm

5. CONCLUSION

The paper explains different central processor scheduling policies in Grid computing environment. We have used cloudsim for building grid computing environment. Priority based scheduling shows lowest makespan which reflect the effectiveness of the priority based scheduling in grid computing. It is very interesting to see the behavior of the proposed priority based algorithm in distributed grid environment where resources are assigned by middleware of the grid environment on the demand of utilization.

REFERENCES

[1] Akash Malhotra, Kunal Gupta, ''A New Framework for Job Scheduling and Resource Management in Grid Environment'', International Journal of Emerging Technology and Advanced Engineering, Vol. 2, Issue. 4, April 2012.

[2] Zafril Rizal M Azmi, Kamalrulnizam Abu Bakar, Mohd Shahir Shamsir, Wan Nurulsafawati Wan Manan, Abdul Hanan Abdullah," Scheduling Grid Jobs Using Priority Rule Algorithms and Gap Filling Techniques", International Journal of Advanced Science and Technology, Vol. 37, December, 2011.

[3] Arindam Das and Ajanta De Sarkar, "On Fault Tolerance Of Resources in Computational Grids", International Journal of Grid Computing & Applications (IJGCA) Vol.3, No.3, September 2012.

[4] S. Gokul Dev, R. Sujith, V.S.Amirutha, S. Divyalakshmi, "An Adaptive Job scheduling methodologies with Fault Tolerance strategy for Computational Grid Environment," International Conference on Computing and Control Engineering (ICCCE 2012), 12 & 13 April, 2012.

[5] Cihan Varol," Dynamic Job Scheduling in Multilayer Grid Networks". Proceedings of the World Congress on Engineering and Computer Science, WCECS 2012, pp. 24-26, Vol. 2 , October, 2012, San Francisco, USA.

[6] Haoqiang Zheng and Jason Nieh, "WARP: Enabling Fast CPU Scheduler Development and Evaluation", IEEE Advanced Computing: An International Journal, pp. 56-59, Vol.3, No.5, September 2009.

[7] Jackson, D., Q. Snell, and M. Clement," Core Algorithms of the Maui Scheduler, in Job Scheduling Strategies for Parallel Processing", D. Feitelson and L. Rudolph, Editors, Springer Berlin / Heidelberg, pp. 87-102, Vol.9, Issue.5, 2001.

[8] Henderson, R.," Job scheduling under the Portable Batch System, in Job Scheduling Strategies for Parallel Processing", D. Feitelson and L. Rudolph, Editors, Springer Berlin / Heidelberg, pp.279-294, Vol.3, No.7, 1995.

[9] Abraham, A., R. Buyya, and B. Nath, Nature's Heuristics for Scheduling Jobs on Computational Grids, in The 8th IEEE International Conference on Advanced Computing and Communications, Cochin, India, 2001.

[10] Garrido, J.M., Performance modeling of operating systems using object-oriented simulation: a practical introduction, Norwell, MA: Kluwer Academic Publishers, 2001.