Peer to Peer Scheduling in PBS and Implementing Multi-Level Feedback Queues

CH.V.T.E.V.Laxmi, K.Somasundaran

Abstract— Grid computing is the federation of pooling resources in order to solve various large problems. Grid computing has gained remarkable importance in the last decade as the resource requirements for large applications increased drastically. Scheduling is the main issue in grid computing and it is the process of making scheduling decisions over multiple grid resources. In this paper we proposed a approach for Portable Batch System (PBS) to implement peer to peer scheduling. This mainly defines that a site can have multiple PBS Pro cluster, each cluster has its server, scheduler and one or more execution systems. This paper focuses on designing of new architecture for Portable Batch System to implement Peer to Peer scheduling system. In this research of Peer to Peer scheduling in PBS we proposed new approach for selecting the jobs from the job pool through ready queue data structure mechanism and we proposed Grid scheduling algorithm Multi Level Feedback Queues Scheduling algorithm for the execution of jobs.

Index Terms— Grid Computing, Grid scheduling, Resource discovery, Portable Batch System, Peer to Peer Scheduling, Pro cluster

I. INTRODUCTION

The Grid is emerging as a new paradigm for solving problems in science, engineering, industry and commerce. Increasing numbers of applications are utilizing the Grid infrastructure to meet their computational, storage and other needs. A single site can simply no longer meet all the resource needs of today's demanding applications, and using distributed resources can bring many benefits to application users. The deployment of Grid systems involves the efficient management of heterogeneous, geographically distributed and dynamically available resources. However, the effectiveness of a Grid environment is largely dependent on the effectiveness and efficiency of its schedulers, which act as localized resource brokers. Figure 1 shows that user tasks, for example, can be submitted via Globus to a range of resource management and job scheduling systems, such as Condor, the Sun Grid Engine (SGE), the Portable Batch System (PBS) and the Load Sharing Facility (LSF). Grid scheduling is defined as the process of mapping Grid jobs to resources over multiple administrative domains. A Grid job can be split into many small tasks.

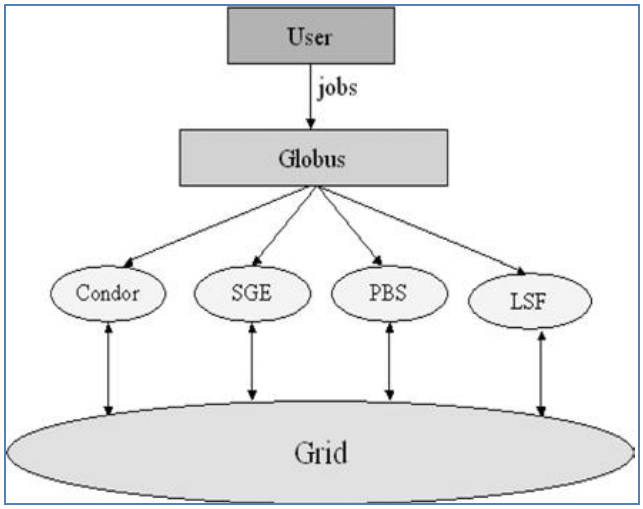

Fig 1.Jobs, via Globus, can be submitted to systems managed by Condor, SGE, PBS and LSF

The PBS is a resource management and scheduling system. It accepts batch jobs (shell scripts with control attributes), preserves and protects the job until it runs; it executes the job, and delivers the output back to the submitter. A batch job is a program that executes on the backend computing environment without further user interaction. PBS may be installed and configured to support jobs executing on a single computer or on a cluster-based computing environment.PBS is capable of allowing its nodes to be grouped into many configurations.

II. RELATED WORK

The PBS architecture as shown in Figure 2, PBS uses a Master host and an arbitrary number of Execution and job Submission hosts. The Master host is the central manager of a PBS cluster; a host can be configured as a Master host and an Execution host.

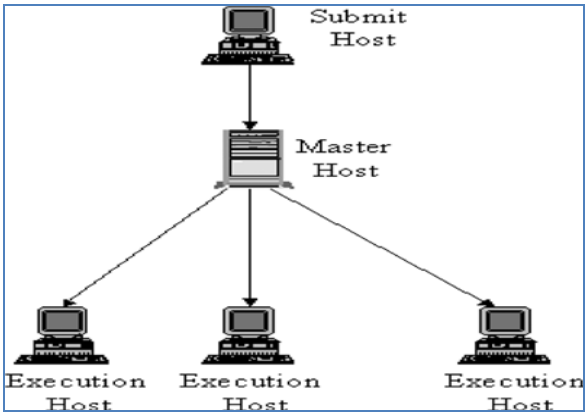

Figure 2 The architecture of a PBS cluster

1. Daemons in a PBS cluster

As shown in Figure 3, to configure a PBS cluster, the following daemons need to be started.

Manuscript Received on July, 2013.

Ch E V T V Laxmi Research Scholar (Karpagam University), Associate Professor, Department of Computer Science Engineering, Raghu Engineering College, Visakhapatnam,Andhra Pradesh,India.

Dr.K.Somasundaram, Research Supervisor(Karpagam University), Professor, Department of Computer Science and Engineering ,Jaya Engineering College,Thiruvallur, Tamilnadu.India.

1. *pbs_server*: The *pbs_server* daemon only runs on the PBS Master host (server). Its main function is to provide the basic batch services, such as receiving/creating a batch job, modifying a job, protecting the job against system crashes and executing the job.
2. *pbs_mom*: The *pbs_mom* daemon runs on each host and is used to start, monitor and terminate jobs, under instruction from the *pbs_server* daemon.
3. *pbs_sched*: The *pbs_sched* daemon runs on the Master host and determines when and where to run jobs. It requests job state information from *pbs_server* daemon and resource state information from *pbs_mom* daemon_ and then makes decisions for scheduling jobs.

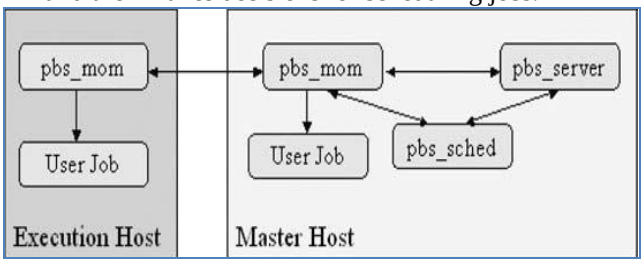

Figure 3 The daemons in a PBS cluster

2. Job selection in PBS

Jobs submitted to PBS are put in job queues. Jobs can be sequential or parallel codes using MPI. A server can manage one or more queues; a batch queue consists of a collection of zero or more jobs and a set of queue attributes. Jobs reside in the queue or are members of the queue. In spite of the name, jobs residing in a queue need not be ordered with FIFO. Access to a queue is limited to the server that owns the queue. All clients gain information about a queue or jobs within a queue through batch requests to the server. Two main queue types are defined: routing and execution queues. When a job resides in an execution queue, it is a candidate for execution. A job being executed is still a member of the execution queue from which it is selected. When a job resides in a routing queue, it is a candidate for routing to a new destination. Each routing queue has a list of destinations to which jobs may be routed. The new destination may be a different queue within the same server or a queue under a different server. A job submitted to PBS can

- be batch or interactive
- define a list of required resources such as CPU, RAM, hostname, number of nodes or any of site-defined resources
- define a priority
- define the time of execution
- send a mail to a user when execution starts, ends or aborts
- define dependencies (such as *after, afterOk afterNotOk or before*)
- be checkpointed (if the host OS has provision for it)
- be suspended and later resumed.

3. Resource matching in PBS

In PBS, resources can be identified either explicitly through a job control language, or implicitly by submitting the job to a particular queue that is associated with a set of resources. Once a suitable resource is identified, a job can be dispatched for execution. PBS clients have to identify a specific queue to submit to in advance, which then fixes the set of resources that may be used; this hinders further dynamic and qualitative resource discovery. Furthermore, system administrators have to anticipate the services that will be requested by clients and set up queues to provide these services.

Additional PBS Pro services include:

- *Cycle harvesting*: PBS Pro can run jobs on idle workstations and suspend or re-queue the jobs when the workstation becomes used, based on either load average or keyboard/mouse input.
- *Site-defined resources*: A site can define one or more resources which can be requested by jobs. If the resource is "consumable", it can be tracked at the server, queue and/or node level.
- *"Peer to Peer" scheduling*: A site can have multiple PBS Pro clusters (each cluster has its server, scheduler and one or more execution systems). A scheduler in any given cluster can be configured to move jobs from other clusters to its cluster when the resources required by the job are available locally.
- *Advance reservations*: Resources, such as nodes or CPUs, can be reserved in advance with a specified start and end time/date. Jobs can be submitted against the reservation and run in the time period specified. This ensures the required computational resources are available when time-critical work must be performed.

III. PROPOSED WORK

The PBS is a resource management and scheduling system. It accepts batch jobs, preserves and protects the job until it runs; it executes the job, and delivers the output back to submitter. As the PBS resource management and scheduling system accepts batch jobs and preserves them unless they are not executed, therefore there is drawback of more waiting time for the submitted jobs. We proposed Peer to Peer scheduling in PBS resource management and scheduling system where a site can have multiple PBS Pro clusters, each cluster has its server, scheduler and one or more execution systems. A scheduler in a cluster can be configured to move jobs from other clusters to its cluster when the resources required by the job are available locally. Fig 4 shows the proposed architecture of PBS Peer to Peer scheduling system.

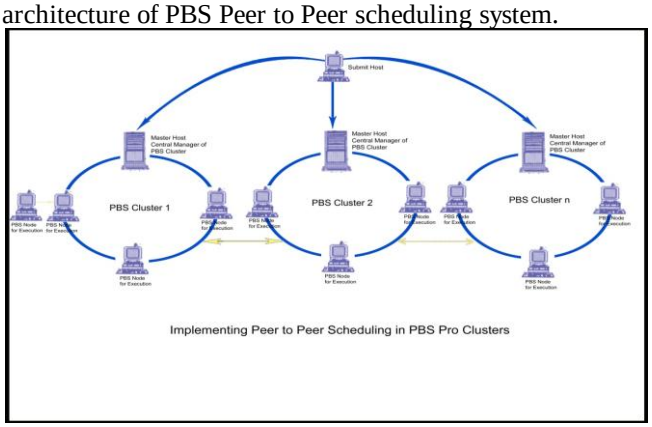

Figure 4 Implementing Peer to Peer Scheduling in PBS Po Clusters

In the above Figure 4 implementing Peer to Peer scheduling in PBS Pro Clusters, we demonstrated and proposed that a site can define one or more execution systems. A scheduler in any given cluster can be configured to move jobs from other clusters to its cluster when then resources required by the job are available locally.

In this paper we also proposed a algorithm i.e. Multi-Level feedback queues algorithm for the job selection in PBS, as the PBS accepts batch jobs, Multi-Level feedback queue algorithm will the best algorithm for job selection process in the PBS resource management and scheduling system.

1. Multi-Level Feedback Queue algorithm

The "Multi-Level Feedback Queue" algorithm allows a process to move between the queues. The idea is to separate process with different CPU-Burst characteristics. If a process uses too much CPU time, it will be moved to next low priority queue. A process that waits too long in a lower priority queue may be moved to a higher priority queue. The main advantage of this algorithm is to prevent starvation.

2. Case Study and Experimental Analysis

For example consider a multilevel feed back queue scheduler with three queues. Numbered RQ0, RQ1, RQ2. The scheduler first executes all process in queue RQ0. Only when RQ0 is empty then it will execute the process RQ1. Similarly the process in RQ2 will only be executed, if RQ0 and RQ1 are empty. A process that arrives for RQ1 will preempts a process in RQ2. A process in RQ1 will turn be preempted by a process arriving for RQ0. Consider the Figure 5.

A process entering in the ready queue is put in queue RQ0. A process in RQ0 is given a time quantum of 16 milliseconds. If it does not finish with in 16 milliseconds, it is moved to the tail of the RQ1. The process in RQ1 execute only if the RQ0 has no process. The process at the head of the RQ1 is given a time quantum of 32 milliseconds. If it does not complete with in the time quantum, it is moved to the tail of the RQ2.

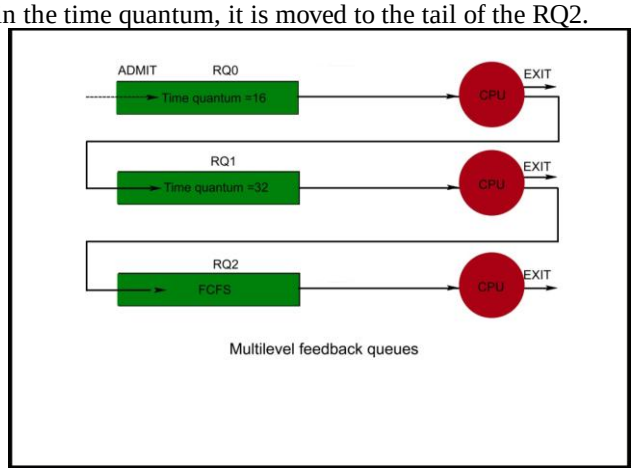

Figure 5 Multilevel Feedback queues

Process in RQ2 are run on an FCFS basis, the process in RQ2 will execute only if the RQ0 and RQ1 are empty. The main advantage of this algorithm is, shorter process will complete quickly, and a longer process need not to wait much time in the ready queue and will gradually draft downward.

IV. CONCLUSION

This paper carries out a survey on grid resource management and scheduling system i.e. Portable Batch System, from the study of different researches carried out on this field. It includes the proposal of Peer to Peer scheduling in the PBS grid resource management and scheduling system. It also demonstrates the proposed architecture of PBS Peer to peer scheduling system. The main aim of this paper is to implement Peer to Peer scheduling in PBS resource management and scheduling system where a site can have multiple PBS Pro clusters each cluster has its server, scheduler and one or more execution systems.

Jobs submitted to PBS are put in job queues and jobs are executed by using proposed algorithm naming Multi- level Feed back Queue algorithm. In this paper the execution of jobs using Multi-Level Feed Back Queue algorithm has been shown with case studies and experimental analysis. PBS, accepts batch jobs, therefore Multi-Level feedback queue algorithm will the best algorithm for job selection process in the PBS resource management and scheduling system.

REFERENCES

1. Ian Foster and Carl Kesselman, "The Grid: Blueprint for a New Computing Infrastructure," Elsevier Inc., Singapore, Second Edition, 2004.
2. Hamscher, V., Schwiegelshohn, U., Streit, A. and Yahyapour, R. Evaluation of Job-Scheduling Strategies for Grid Computing. GRID 2000, 191–202, 17–20 December 2000, Bangalore, India. Lecture Notes in Computer Science, Springer-Verlag.
3. PBS Pro, http://www.pbspro.com/.
4. Cactus, http://www.cactuscode.org/. 300 GRID SCHEDULING AND RESOURCE MANAGEMENT
5. PBS Pro, http://www.pbspro.com
6. Edi.Laxmi Scheduling in Grid Computing International Journal of Computer Science and Management Research Vol 2 Issue 1 January 2013
7. Goux, Jean-Pierre, Kulkarni, Sanjeev, Yoder, Michael and Linderoth, Jeff. Master-Worker: An Enabling Framework for Applications on the Computational Grid. Cluster Computing, 4(1): 63–70 (2001).
8. Cactus, http://www.cactuscode.org/. 300 GRID SCHEDULING AND RESOURCE MANAGEMENT Spooner, D., Jarvis, S., Cao, J., Saini, S. and Nudd, G. Local Grid Scheduling Techniques using Performance Prediction, IEE Proc. – Comp. Digit. Tech., 150(2): 87–96 (2003
9. Enterprise Edition policy,

AUTHOR PROFILE

Ch E V T V Laxmi Research Scholar (Karpagam University), Associate Professor, Department of Computer Science Engineering,, Raghu Engineering College, Visakhapatnam,Andhra Pradesh,INDIA.She is Microsoft Certified Solution Developer.(MCSD).
email:elaxmi2002@yahoo.com.

Dr.K.Somasundaram, Research Supervisor(Karpagam University), Professor, Department of Computer Science and Engineering ,Jaya Engineering College,Thiruvallur , Tamilnadu.INDIA
email:soms72@yahoo.com.