A Short Survey of Commercial Cluster Batch Schedulers

Yoav Etsion Dan Tsafrir
School of Computer Science and Engineering
The Hebrew University, 91904 Jerusalem, Israel

Abstract

As high performance computing clusters are getting cheaper, they become more accessible. The various clusters are running a host of workload management software suites, which are getting more complex and offer cluster administrators numerous features, scheduling policies, job prioritization schemes, etc.

In this paper we survey some of the common commercial workload managers on the market, covering their main features — specifically the scheduling policies and algorithms they support, their priority and queueing mechanisms, focusing on their default settings.

Introduction

High performance computing clusters are getting a common commodity in scientific oriented companies and research institutions. This trend is accompanied by the introduction of commercial cluster management software suites, handling all aspects of cluster resource allocations.

These software suites offer cluster administrators a plethora of features and tunable parameters. These include applications' queue management policies, process prioritization, and scheduling algorithms.

There is no viable data about the extent to which cluster administrator tune the management software's configuration from its default values. To the best of our knowledge though, most of these high performance computing sites administrators do not stray far from the default configuration of their workload management software — if they even stray from it at all. It seems that only in rare cases do the administrators use the entire spectrum of tunable parameters.

In this paper we present a survey of some common commercial workload managers, focusing on the features they offer, scheduling policies they support, focusing on their default settings.

1 Moab/Maui

The *Moab Workload Manager* [1] is based on the Maui batch scheduler [8], with all its flexibility and added features – backfilling, service factors, resource constraints and weights, fair-share options, direct user/group/account prioritization, target wait times, etc.

However, based on the Maui Scheduler Administrator's Guide [4] its default behavior out of the box is a simple FCFS batch scheduler, with a backfilling policy that maintains a time reservation for the first job in the queue – EASY backfilling. We have verified that fact in the source code [2] of the job priority function (*MJobGetStartPriority(...)* in *MPriority.c*):

In the Maui scheduler, the priority of each job is a weighted sum of several factors, where the weights are set by the administrator. Each factor itself is a weighted sum of sub-factors, whose weights, again, are governed by the administrator. After looking at the source code we found that even though all the factor's weights are set to 1 (in an array called *CWeight*), all the weights of the sub-factors are set to 0, except for that of the job's queue time which is set to 1 (all the sub-factors weights are saved in the *SWeight* array). The result is that the job's queue time is the only factor that is not zero, and even though the factor weights are set to 1, the queue time is the priority function — resulting in a FCFS scheduler.

2 LoadLeveler

IBM's *LoadLeveler* [9] supports several schedulers, such as FCFS, FCFS with backfilling, gang scheduling, and can also interface with external schedulers. The system also supports checkpointing and restarting of running jobs, and specific *IBM SP* hardware.

Within its own set of schedulers many of the features are tunable: first and foremost, an administrator can rewrite the priority function *SYSPRIO* and use current system data. Examples for such data are a user's class, how many jobs the user/group has in the system, etc. Other parameters can also be used to establish a fair-share priority function. The administrator can also set specific privi-

leged user/group/class accounts. This, coupled with support for job preemption, allows for high priority jobs to preempt low priority ones. At the user level, each user can change the running order (or explicitly specify one) of his own jobs.

Loadleveler supports backfilling, and can even can be tuned to use either the best-fit or first-fit metrics to choose jobs for backfilling.

The default scheduling of LoadLeveler is FCFS: the default priority function is FCFS, as the *SYSPRIO* function is simply the job's queue time. Backfilling is *not* set by default, but when enabled, its policy is first-fit, with time reservation set only for the first job in the queue When using backfilling, users are obligated to specify a runtime estimate for their jobs. When a job exceeds its time estimate, it is killed (sent a SIGKILL signal). This is similar to the EASY backfilling policy.

3 Load Sharing Facility (LSF)

Platform's *Load Sharing Facility* (*LSF*) [11, 12] is a comprehensive solution for high performance workload management. It supports numerous scheduling algorithms, including FCFS, fair-share, preemptive, backfilling and Service Level Agreements level (SLA). LSF can also interface with external schedulers such as Maui. Other features include system supports for automatic and manual checkpoints, migrations, automatic job dependencies and job reruns.

The fair-share scheduler lets the administrator assign shares to users and groups, and set a priority function that divides the resources according to the assigned shares. The shares can also be assigned in a hierarchical manner, so a group can be assigned shares, and divided into subgroups, each getting a percentage of the shares. The final priority function of the fair-share scheduler takes into account the standing shares for the user (either directly of via his group), and the number of running and queued jobs he has.

The Service Level Agreements (SLA) scheduler is a high level scheduler that allows the administrator to state a goal for the system — job deadlines, throughput etc. — without having to tune the lower levels of the scheduler for achieving that goal.

An interesting scheme, called *priority escalation* is also introduced in this software suite. In this scheme, the administrator can set an escalating parameter on a job's priority, so it's priority will increase every time interval – giving much higher priorities to waiting jobs, even when using a fair share scheduler.

Jobs are submitted to queues with different priorities, which the administrator defines. He can also define different scheduling schemes for each queue. Both preemption and backfilling are considered to be queue properties: a queue can be declared to be preemptive, in which case its jobs can preempt running jobs from any lower level queue that is set to be preemptible. Backfilling can be turned on for a queue in the queue configuration file. This flag is not set by default, but the default behavior of processor reservation is similar to EASY – reserve the processor for the first job in the queue when backfilling. Each backfilling queue is assigned a job time limit, which is used if the user did not specify a time limit upon submission. Backfilling queues have their limitations – backfilled jobs cannot be preemptible, as they would consume resource reserved for another job.

If the administrator does not define any queues, a default queue is used, and its scheduling is set to FCFS. The administrator guide [12] is careful to warn that this policy might not be best, and that the site administrator must take that into consideration and define special queues. As mentioned previously, backfilling is not enabled by default, but when enabled, it's default behavior is similar to EASY.

4 Portable Batch System (PBS)

The *Portable Batch System* (PBS) comes in two flavors: *OpenPBS* [16] is intended for small clusters, and *PBS-Pro* [3] is the full fledged, industrial strength version (both are descendants of the system described in [7]). For simplicity, we will focus on PBS-Pro.

The suite includes a very versatile scheduler support. Schedulers included with the suite are FCFS, SJF, user/group priorities and fair-share. Also, site specific schedulers can be implemented natively in the C and TCL programming languages, or in a special language called BaSL. Other features include checkpoint support, re-pack and rerun support for failed or stopped jobs, and failed nodes recovery.

The fair-share scheduler uses a similar hierarchical approach, similar to *LSF*. An administrator can distribute shares among groups, whose shares can, in turn, be divided to subgroups. This creates a tree structure in which each node is given shares, which are distributed by administrator assigned ratios to its child nodes, all the way down to the tree leaves. The leaves themselves can be either groups or specific users.

As with other software suites, the administrator can define work queues with various features. Queues can have certain resource limits that are enforced on the jobs they hold. A job can even be queued according to its specified resource requirements — the administrator can define a queue for short jobs, and the queuing mechanism can automatically direct a job with small CPU requirements to the short jobs queue. Of course the administrator can de-

fine a priority for each queue, thus setting the dispatch order between queues, or can be selected for dispatch in a round robin fashion. Queues can also be set inactive for certain times, which allow using desktops as part of the cluster at night or holidays.

The *PBS-Pro* system support preemption between different priority jobs. An administrator can define a preemption order between queues, by which jobs from higher priority queues can preempt jobs from lower priority queues if not enough resources are available. Inter-queue preemption is enabled by default, but there is only one default queue.

Being the exception that makes the rule, the default scheduler in both PBS systems is SJF. To prevent starvation (which is the main problem of SJF scheduling), the system can declare a job as starving after some time it has been queued (with the default time set to 24 hours). A starving job has a special status — no job will begin to run until it does. The result begin is that declaring a job as starving causes the system to enter a draining mode, in which it lets running jobs finish until enough resources are available to run the starving job. The starvation prevention mechanism can be enabled only for specific queues.

Backfilling is supported, but only in context of scheduling jobs around a starving job waiting to run, and only if users specify a wall time CPU limit. Like the starvation prevention mechanism, backfilling can also be enabled for specific queues.

As mentioned before, the default scheduler is SJF, and both the starvation prevention mechanism and backfilling enabled for all queues.

5 Sun Grid Engine (SGE)

The *Sun Grid Engine* (SGE) [15, 13, 14] is much simpler then its contenders.

SGE has two scheduling policies: FCFS, and an optional administrator set function of Equal-Share scheduler. The latter is a simple fair-share scheduler that tries to distribute resources equally among all users and groups. For example, to overcome a case where a user submits many jobs over a short period of time, its latter jobs will be queued until other users had a chance to run their jobs.

An administrator can also define new job queues, with specific dispatch order among the queues themselves.

Currently, the system does not support backfilling, although this features is planned to be incorporated in future versions [6].

The default behavior is still FCFS, since the default priority function is again the job's queue time.

6 OSCAR

During this survey we have also enquire about *OSCAR* [5], which is sometimes regarded as a workload management software suite. However, this is more of a cluster installation software, which helps manage the nodes belonging to the cluster, assign them IP addresses, network mounted root file systems, and other resources. The workload management itself is done using one of the aforementioned software suites, mainly *Maui* or *OpenPBS*.

Conclusion

In this paper have surveyed some of the more prominent cluster management software suites, trying to describe the features they offer. As we suspect most cluster administrators do not stray far from the different suites' default configurations, we have focused on describing those for the various suites.

To our surprise, we found that the prevalent default scheduler setting is FCFS, and in those management suites that also support backfilling, the governing scheme used is EASY [10]. If the prominent software suites' configuration is indeed their default one, it seems most cluster use the very simplistic FCFS scheduling algorithm, or in the more complex systems — EASY scheduling.

References

[1] MOAB workload manager. http://www.supercluster.org/moab/.

[2] MOAB workload manager (Maui scheduler) source code. http://www.supercluster.org/moab/. Version 3.2.6.

[3] Altair Grid Technologies. *PBS Pro Administrator Guide 5.4*, 2004. Editor: James Patton Jones.

[4] Cluster Resources, Inc. *Maui Scheduler Administrator's Guide*. Version 3.2.

[5] B. des Ligneris, S. Scott, T. Naughton, and N. Gorsuch. Open source cluster application resources (OSCAR): design, implementation and interest for the [computer] scientific community. In *First OSCAR Symp.*, May 2003.

[6] A. Haas. Reservation / preemption / backfilling in grid engine 6.0. In *2nd Grid Engine Workshop*. Sun Microsystems GmbH, Sep 2003.

[7] R. L. Henderson. Job scheduling under the Portable Batch System. In D. G. Feitelson and L. Rudolph,

editors, *Job Scheduling Strategies for Parallel Processing*, pages 279–294. Springer-Verlag, 1995. Lect. Notes Comput. Sci. vol. 949.

[8] D. Jackson, Q. Snell, , and M. Clement. Core algorithms of the Maui scheduler. In *Job Scheduling Strategies for Parallel Processing*, Lecture Notes in Computer Science. Springer-Verlag, 2001.

[9] S. Kannan, M. Roberts, P. Mayes, D. Brelsford, and J. F. Skovira. *Workload Management with LoadLeveler*. IBM, first edition, Nov 2001. ibm.com/redbooks.

[10] D. Lifka. The ANL/IBM SP scheduling system. In D. G. Feitelson and L. Rudolph, editors, *Job Scheduling Strategies for Parallel Processing*, pages 295–303. Springer-Verlag, 1995. Lect. Notes Comput. Sci. vol. 949.

[11] Platform Computing Inc. Platform LSF. http://www.platform.com/products/LSFfamily/.

[12] Platform Computing Inc. *Administering Platform LSF*, Jan 2004. www.platform.com/services/support/docs_home.asp.

[13] Sun Microsystems, Inc. *Sun ONE Grid Engine Enterprise Edition Administration and User's Guide*, 2002. version 5.3.

[14] Sun Microsystems, Inc. *N1 Grid Engine 6 Administration Guide*, 2004.

[15] Sun Microsystems, Inc. Sun grid engine. http://gridengine.sunsource.net/, 2004.

[16] V. Systems. *Portable Batch System, Administrator Guide*, 2000. OpenPBS Release 2.3.