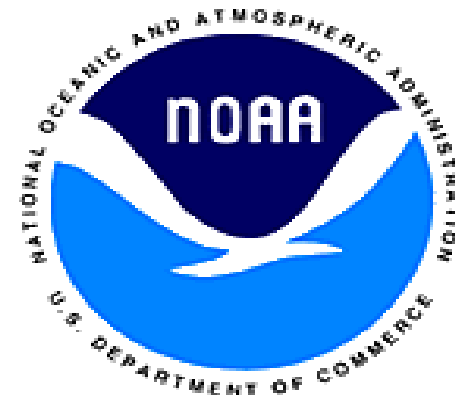
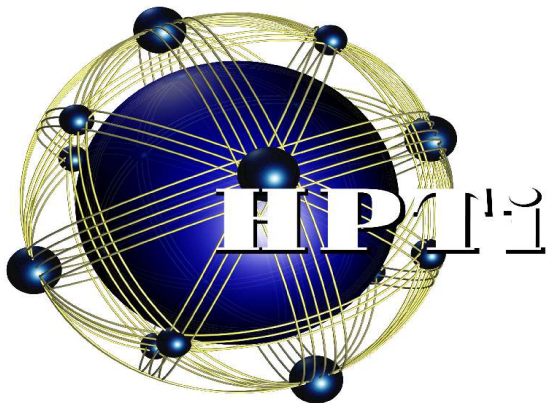


Scheduling and Job Management using Grid Engine on a Multi-Teraflop HPC

Craig Tierney

September 22-24, 2003
Regensburg, Germany



Forecast System Laboratory – NOAA

- # Mission is to research and develop better forecasting technologies; instrumentation, modelling, and methods to incorporate both.
 - # In 1999, 5 year, \$15 million USD contract for new supercomputing resources
 - # High Performance Technologies Inc. (HPTi) was chosen to deliver Linux cluster, Jet, as their HPC system
 - # First time that a Linux cluster was purchased through competitive bid by the US government
-

High Performance Technologies Inc.

- # Reston, Virginia USA Company
 - # 50 fastest growing companies in Virginia
 - # Specialize in cutting edge technology
 - Linux HPC clusters (design, implementation, programming)
 - Re-programmable computing (FPGA)
 - Deployable cluster computing
 - Large scale hardware/software integration
 - Enterprise Architecture
-

Jet History

- # Compute system was delivered in 3 phases
 - # First phase, March 2000
 - 276 Compaq XP1000s (667 Mhz, 512MB per processor, Myrinet 1280)
 - # Second phase, September 2001
 - Added 142 Dual API UP2000 (833 Mhz, 512 MB per processor, Myrinet 2000)
-

Jet History, Cont.

- # First 2 phases used OpenPBS
- # OpenPBS provided robust features to provide batch queuing for a large cluster
- # Poor communication model and server design resulted in poor stability and scalability
- # The final system was to be 3-5x larger than the initial systems, there was a concern that OpenPBS would not scale to this size.

3rd Phase – Intel Jet

Compute nodes based on Intel Xeon processor

Better price and availability than Alpha systems

Better price/performance than Alpha

Better Linux support

Installed in 2 sections

128 nodes delivered in July 2002 (2.2 Ghz Xeon, 512 MB per CPU, Myrinet 2000 Fibre)

Initial XP1000 system removed

640 more nodes delivered in September 2002

All 768 nodes were put into production in November 2002.

3.337 Tflops, 8th fastest on Top500 list, Nov. 2002

FSL System Usage

- # Resources are divided between organizations.
 - 40% Internal, 40% NCEP, 20% external users
 - # Jobs can be classified as real-time or research.
 - # Real-time jobs are run regularly and have time constraints in which to complete. For many projects, data are delivered offsite and have deadlines.
 - # Large jobs and long run times are not the norm. Average job is 8-64 cpus, 0.5-4 hours.
 - # Must be able to perform system maintenance on portions of the system while important real-time jobs (RUC20, LAPS) still complete.
-

Batch Queuing System Needs

- # Besides stable, robust, redundant.....
 - # Support multiple disconnected Myrinet systems
 - hide details from the users
 - # Schedule only 1 job per SMP node
 - # User/account/class controls over
 - max cpus per job
 - max wallclock per job
 - max cpus running at one time
 - max jobs
 - queue accessibility
-

Batch Queuing System Needs, Cont.

- # Control placement of rank 0 node (IO node),
mimic OpenPBS complex resource node
allocation (-lnodes=1:io+N:comp)
 - # Overflow queue
 - # Resource reservation
 - # Compatibility with OpenPBS syntax
-

Why Grid Engine?

- # Stability, Reliability, Scalability

- # Features

 - Failover/shadow server

 - Parallel environment/MPI job support

 - Consumable resources

 - Multi-platform support

- # Open Source

 - Free (as in beer) was NOT the most important feature

Additional Features

- # Grid Engine did not support all features needed

No batch system did (LSF, PBSPro)

- # Features were implemented in 3 pieces

Wrapper to qsub

Grid Engine pre-scheduler (sge_preschedd)

Patch to qstat

Qsub wrapper

Qsub wrapper was necessary to

Verify job maximums and queue access

Verify user membership in specified account for system accounting

Translate OpenPBS syntax

SGE Prescheduler

- # Modify job parameters so only one job is scheduled per node
 - # Fit job on one of multiple disconnected networks (separate Myrinet networks)
 - # Release job only when account maximums for user/account/class are not exceeded
 - # IO node support
-

Pre-scheduler Configuration

```
# Setup maps for compute nodes
map comp qcomp2 qcomp3 qcomp4 qcomp5
map comp1 qcomp1
map comp2 qcomp2
map comp3 qcomp3
map comp4 qcomp4
map comp5 qcomp5
```

```
# Make mapping for PVFS
map pvfs qcomp3 qcomp4 qcomp5
```

```
# Setup IO nodes
map io qio3 qio41 qio42
map io3 qio3
map io41 qio41
map io42 qio42
```

```
io qio3 g0128.q
io qio41 g0256.q
io qio42 g0257.q
```

IO Node support

- # Some nodes have faster IO connectivity (GigE) or more memory
 - # Several codes need rank 0 node to have additional memory
 - # Several codes with high IO requirements that do IO through a single node
 - # Using *-masterq* allows selection of the rank 0 node, but not user friendly
 - # User specifies special PE (io) and consumable resource (*-l io*) to gain access to IO node
-

What about Maui?

- # Wasn't supported
- # Maui is a very powerful/robust scheduler, but scheduling features weren't needed
 - All that is used is FIFO with priorities
- # Still needed the pre-scheduler to ensure 1 job per node and overflow queue
- # Maui did not support max per user/account/class resources

Ok to submit jobs above maximums, just shouldn't run.

Qstat Patch

- # New feature to qstat (-c) which displays jobs in a easier to read format when only parallel environment jobs are run.
 - # Patch written by James Vasak of HPTi.
-

Qstat Patch, Cont.

Job-ID	Jobname	Username	Account	State	Cpus	Queue	Time	Time
1033321	Mvkf84091011.NRII	lili	of_ciaqex	r	16	qcomp2	22:31	24:00
1036621	Mv1988070809.NNRP	lili	of_ciaqex	r	16	qcomp2	10:30	24:00
1037715	ccm3.10	shin	eab	r	16	qcomp2	05:45	08:00
1037993	ccm3.10	shin	of_crotc	r	16	qcomp2	03:33	08:00
1037994	ccm3.10	shin	of_crotc	r	16	qcomp2	03:33	08:00
1037995	ccm3.10	shin	of_crotc	r	16	qcomp2	02:36	08:00
1038001	ccm3.10	shin	of_crotc	r	16	qcomp4	03:32	08:00
1038011	ccm3.10	shin	of_eab	r	16	qcomp4	03:30	08:00
1038012	ccm3.10	shin	of_eab	r	16	qcomp4	03:30	08:00
1038185	wrf	harrop	of_jetmgmt	r	64	qcomp4	01:54	06:00
1038191	wrf	harrop	of_jetmgmt	r	64	qcomp4	01:52	06:00
1038432	wrf	harrop	of_jetmgmt	r	64	qcomp4	01:03	06:00
1038659	Mv198904.NRII	hengliu	of_ciaqex	r	32	qcomp3	00:32	24:00
1038662	retro_wrf	rt-aq	of_ap-fc	r	64	qcomp5	00:32	06:00
1038772	amie19990406	ridley	swr	r	8	qcomp4	00:02	06:00
1038773	amie19990407	ridley	swr	r	8	qcomp3	00:01	06:00
1038783	amie19990417	ridley	swr	r	8	qcomp4	00:01	06:00
1038784	amie19990418	ridley	swr	qw	8	comp	--:--	06:00
1038785	amie19990419	ridley	swr	qw	8	comp	--:--	06:00
1038789	amie19990423	ridley	swr	qw	8	comp	--:--	06:00
1038790	amie19990424	ridley	swr	qw	8	comp	--:--	06:00

Grid Engine Configuration

- # Master/shadow fail-over configuration

 - server nodes mount \$SGEROOT from NFS server (/home)

- # All other nodes have local \$SGEROOT.

- # Link to NFS server for act_qmaster for fail-over support

- # All queues are the same

- # Each server runs sge_preschedd (supports fail-over)

Intel Jet Statistics

- # 772 nodes configured in SGE in six separate parallel environments
 - 5 systems connected with separate myrinet
 - 1 system (4 nodes) used for visualization (no myrinet)
 - # Running ~320 days since acceptance
 - # ~60 days since last failover
 - Servers haven't been a problem since kernel bug was fixed approximately 6 months ago.
 - # Approximately 1.1 million jobs run
 - # Approximately 3400 jobs per day
-

Alpha Jet Statistics

- # 142 nodes across two separate myrinet networks
 - # Approx. 220 days since SGE conversion
 - # Approx. 90 days since last failover
 - # Approx. 410,000 jobs run
 - # There had been issues with the SGE servers running on Alpha. We thought it was more likely a system/OS issue than SGE. The SGE servers were moved to Intel nodes. No problems since.
-

Code Availability

- # Code is open source (BSD or public domain, not GPL) still trying to work with government lawyers
 - # qsub required rewrite
 - original script called OpenPBS code to translate syntax
 - Finished but needs more testing
 - # Email ctierney@hpti.com for code/information, or watch the [gridengine-users](#) list for information.
-

Todo

- # Add Qbank support (really an SGE issue)

- # Add reservations

Did this before for OpenPBS. Easy to do, hard to do right.

- # Add checks in `sge_preschedd` to prevent 'accidental' circumvention of system by users.

- # Add support for single thread jobs, ignore 1 job per node rule for set of nodes.

- # Have Grid Engine just do all this automatically

Conclusion

- # Grid Engine has been used over the last year to provide batch queuing facilities for two HPC Linux clusters (Intel and Alpha) at FSL
 - # It is a very robust, stable, and reliable base to provide batch scheduling to users
-