Job Scheduler Details

http://www.accre.vanderbilt.edu
Outline

- Overview of the Cluster Scheduler Environment (slide 3)
- TORQUE/PBS Resource Manager (slide 4)
- Maui/Moab Job Scheduler (slide 5)
- TORQUE ↔ Moab Coordination (slide 6)
- Job Submission Review (slides 8-9)
- The Queue (slide 7)
- More Scheduler Commands (slides 10-12)
- Checking Cluster Status (slide 13)
- Scheduler Etiquette, Policies, Memory Usage (slides 14-16)
- Staging Jobs (slides 17-18)
- Using PBS Variables (slides 19-21)
Scheduler Queueing System

- Software suites:
  - TORQUE/PBS – resource manager
  - Maui/Moab – job scheduler

- Queue position and wait times depend on:
  - Fairshare – Percentage of CPUs
  - Priority – Calculated based on fairshare and queuetime
TORQUE Resource Manager

- Tera-scale Open-source Research and QUEue manager
- Built upon original Portable Batch System (PBS) project
- Resource manager: Manages availability of, and requests for, compute node resources
Moab Scheduler

- Job scheduler
- Implements and manages:
  - Scheduling policies
  - Dynamic priorities
  - Reservations
  - Fairshare
Sample Job Flow

- Script submitted to TORQUE (using `qsub`) specifying required resources
- Moab periodically retrieves from TORQUE list of potential jobs, available node resources, etc.
- Moab prioritizes jobs in idle queue
- When resources become available, Moab tells TORQUE to execute certain jobs on particular nodes
- TORQUE dispatches jobs to the PBS MOMs (machine oriented miniserver) running on the compute nodes - `pbs_mom` is the process that starts the job script
- Job status changes reported back to Moab, information updated
- Moab sends further instructions to TORQUE
- Moab updates occur roughly every 15 minutes
The Queue (review)

- Queue divided into 3 subqueues:
  - active - running
  - eligible - idle, but waiting to run
  - blocked - idle, held, deferred

- A job can be “blocked” for several reasons, e.g.,
  - requested resources not available
  - reserved nodes offline
  - user has maximum of 10 jobs in eligible queue
  - user places intentional hold
  - Moab supports four distinct types of holds, *user* holds, *system* holds, *batch* holds, and *defer* holds
Job Submission (review)

- **qsub**

  ➢ **Example script: sample.pbs**

    ```
    #!/bin/tcsh
    #PBS -M mail.address@vanderbilt.edu
    #PBS -m bae
    #PBS -l nodes=4:x86:myrinet
    #PBS -l walltime=00:05:00
    #PBS -l mem=500mb
    #PBS -o myjob.output
    echo "The sample job is beginning."
    ... job control ...
    echo "The sample job is done."
    ```

  ➢ **qsub sample.pbs**
Job Submission (review)

- Cluster specific PBS node attributes (#PBS -l)
  - ppc64, nomyrinet
  - ppc64, myrinet
  - x86, p4, nomyrinet
  - x86, p4, nomyrinet, imagic
  - x86, p4, nomyrinet, imagic, bigmem
  - x86, opteron, nomyrinet
  - x86, opteron, nomyrinet, bigmem
  - x86, opteron, nomyrinet, dualdual
  - x86, opteron, myrinet
  - x86, opteron, myrinet, twogig

- E. g., #PBS -l nodes=4:x86:myrinet:twogig

- Maximize resource pool
- Leave \textit{walltime} and \textit{mem} buffer
Checking Job Status (review)

- TORQUE/PBS and Moab scheduler and job submission documentation at Cluster Resources:


- Help for specific commands:
  - Under TORQUE Resource Manager follow these links:
    - TORQUE Wiki Documentation
    - Documentation overview
    - A. Commands overview
  - Under Moab Workload Manager follow these links:
    - Commands Documentation
Checking Job Status (review)

- Useful TORQUE/PBS commands (**no man page on cluster)**
  - `pbsnodes`
  - `qalter`
  - `qdel`
  - `qhold`
  - `qrerun`
  - `qstat`
  - `qsub`
  - `tracejob**`
Checking Job Status (review)

- Useful Moab commands (**no man page on cluster)
  - `checkjob -v`
  - `checknode**`
  - `mdiag** -f and -p and -j` (previously `diagnose`)
  - `mjobctl**`
  - `showq`
  - `showres`
  - `showstart**`

- `http://www.accre.vanderbilt.edu/support/selfhelp/faq.php`
Checking Utilization

- Utilization charts on website:
  - Overview charts:
    - Number and percentage of active compute CPUs
    - Number and percentage of active compute nodes
    - Number of active, eligible, and blocked jobs
  - Utilization by CPU and connectivity type:
    - Number of active PPC’s, Opteron’s, and P4’s
    - Subset of active PPC’s and Opteron’s with Myrinet

http://www.accre.vanderbilt.edu/utilization/index.php
Scheduler Etiquette

- Our goal is to provide fair use of the resources
  - 100% fair usage
  - Set number of CPUs becoming free every hour
- You should *stage* large quantity job submissions
- You should maximize use of available resources
  - Plan submission around slow times, if possible
  - Plan ahead for long jobs
- Talk to us to arrange special needs
Scheduler Etiquette

- Scheduler policies
  
  http://www.accre.vanderbilt.edu/mission/cluster_policies/job_scheduler.php

- Limits on:
  - Number of jobs in queue
  - Maximum and minimum job lengths
  - Memory Usage
  - Running on Myrinet nodes

- We place special restrictions when necessary
Scheduler Etiquette

- Monitoring memory usage (until new version of Moab):
  - Use `linux pmap` on node to estimate memory usage of running job

  http://www.accre.vanderbilt.edu/support/selfhelp/check_job.php

  - Use `p_reaper` in your PBS script to auto-kill jobs that cause memory problems, see:

    accre-forum 2007 March archive
Job Dependencies

- Can use to *dependencies* to submit consecutive jobs, e.g.,

  ```
  qsub -W depend=afterok:jobid
  ```

- When would you use this?
  - Job walltime > 30 days
  - Stitching together multiple programs
  - But programming in script usually a better approach. (Why?)
Advanced topics

- **Job staging, plus demonstration**
- **Using PBS environment variables, plus demonstration**
- **Job stdout/stderr - simple examples on web - qcat, plus demonstration**
Using PBS Variables

- When a batch job starts, variables are introduced into the compute node’s environment which can be used by batch scripts to make decisions, create output files, etc.

<table>
<thead>
<tr>
<th>variable</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>PBS_JOBCOOKIE</td>
<td>job cookie</td>
</tr>
<tr>
<td>PBS_JOBNAME</td>
<td>user specified jobname</td>
</tr>
<tr>
<td>PBS_MOMPORT</td>
<td>active port for mom daemon</td>
</tr>
<tr>
<td>PBS_NODEFILE</td>
<td>file with list of allocated nodes</td>
</tr>
<tr>
<td>PBS_NODENUM</td>
<td>node offset number</td>
</tr>
<tr>
<td>PBS_O_HOME</td>
<td>homedir of submitting user</td>
</tr>
</tbody>
</table>

- Continued next slide:
Using PBS Variables

- Continued...

<table>
<thead>
<tr>
<th>variable</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>PBS_O_HOST</td>
<td>host of currently running job</td>
</tr>
<tr>
<td>PBS_O_JOBID</td>
<td>unique PBS job ID</td>
</tr>
<tr>
<td>PBS_O_LANG</td>
<td>language variable for job</td>
</tr>
<tr>
<td>PBS_O_LOGNAME</td>
<td>name of submitting user</td>
</tr>
<tr>
<td>PBS_O_PATH</td>
<td>path to executables used in job script</td>
</tr>
<tr>
<td>PBS_O_SHELL</td>
<td>script shell</td>
</tr>
<tr>
<td>PBS_O_WORKDIR</td>
<td>jobs submission directory</td>
</tr>
<tr>
<td>PBS_QUEUE</td>
<td>job queue</td>
</tr>
<tr>
<td>PBS_TASKNUM</td>
<td>number of tasks requested</td>
</tr>
</tbody>
</table>
Example Script