Submitting and Running Jobs

Scott French
NERSC User Services Group

New User Training
February 23, 2015
Jobs at NERSC

• Most jobs are parallel, using 10s to 100,000+ cores
• Production runs execute in batch mode
• Interactive and debug jobs are supported for up to 30 minutes
• Typically run times are a few to 10s of hours.
  – Each machine has different limits.
  – Limits are necessary because of MTBF and the need to accommodate 5,500 users’ jobs
• Also a number of ‘serial’ jobs
  – Typically some kind of pleasantly parallel simulation
Login Nodes and Compute Nodes

- Each supercomputer has 3 types of nodes visible to users
  - Login nodes
  - Compute nodes
  - Application launcher or “MOM” nodes

- Login nodes
  - Edit files, compile codes, run UNIX commands
  - Submit batch jobs
  - Run short, small utilities and applications

- Compute nodes
  - Execute your application; dedicated to your job
  - No direct login access

- Application launcher or “MOM” nodes
  - Execute your batch script commands
  - Carver: “head” compute node
  - Edison / Hopper: shared “service” node (not a compute node)
Launching Parallel Jobs

Login Node

MOM Node

Compute Nodes

qsub

aprun / mpirun
Launching Parallel Applications

• An “application launcher” executes your code
  – Distributes your executables to all your nodes
  – Starts concurrent execution of N instances of your program
  – Manages execution of your application
  – On Edison / Hopper: the launcher is called “aprun”
  – On Carver: “mpirun”

• Only the application launcher can start your application on compute nodes

• You can’t run the application launcher from login nodes
Submitting Batch Jobs

• To run a job on the compute nodes you must write a “batch script” that contains
  – Batch directives to allow the system to schedule your job
  – An aprun or mpirun command that launches your parallel executable

• Submit the job to the queuing system with the qsub command
  – `qsub my_batch_script`
Edison - Cray XC30

- 133,824 cores, 5,576 nodes
- “Aries” interconnect
- 2 x 12-core Intel ’Ivy Bridge' 2.4 GHz processors per node
- 24 processor cores per node, 48 with hyperthreading
- 64 GB of memory per node
- 357 TB of aggregate memory
- 2.7 GB memory / core for applications
- /scratch disk quota of 10 TB
- 7.6 PB of /scratch disk
- Choice of full Linux operating system or optimized Linux OS (Cray Linux)
- Intel, Cray, and GNU compilers
```bash
#PBS -q debug
#PBS -l mppwidth=96
#PBS -l walltime=00:10:00
#PBS -N my_job

cd $PBS_O_WORKDIR
aprun -n 96 ./my_executable
```
#PBS -q debug
#PBS -l mppwidth=96
#PBS -l walltime=00:10:00
#PBS -N my_job

cd $PBS_O_WORKDIR
aprun -n 96 ./my_executable

Job directives: instructions for the batch system
- Submission queue
- How many compute cores to reserve for your job (/ 24 = # nodes)
- How long to reserve those nodes
- Optional: what to name STDOUT files, what account to charge, whether to notify you by email when your job finishes, etc.
#PBS -q debug
#PBS -l mppwidth=96
#PBS -l walltime=00:10:00
#PBS -N my_job

cd $PBS_O_WORKDIR
aprun -n 96 ./my_executable

Change from home directory to job submission directory
- Script is initially run from your home directory, **which is not advisable** (as we mention in the filesystem intro)
- You will see much better performance if your job reads / writes from one of the high-performance scratch filesystems
Sample Edison Batch Script - MPI

#PBS -q debug
#PBS -l mppwidth=96
#PBS -l walltime=00:10:00
#PBS -N my_job

cd $PBS_O_WORKDIR

aprun -n 96 ./my_executable

Launches parallel executable on the compute nodes

• Carries over (partial) login environment
• Controls how your executable:
  • maps to processors on the compute nodes (e.g. how many tasks?)
  • accesses the memory on each processor
Sample Edison Batch Script - MPI

```bash
#PBS -q debug
#PBS -l mppwidth=96
#PBS -l walltime=00:10:00
#PBS -N my_job

cd $PBS_O_WORKDIR
aprun -n 96 ./my_executable
```

mppwidth is number of compute cores requested for your job
- mppwidth = 24 x # of nodes on Edison (and Hopper)
- must be greater than or equal to the number of tasks requested (\(-n\)
Sample Edison Batch Script - MPI

#PBS -q debug
#PBS -l mppwidth=192
#PBS -l walltime=00:10:00
#PBS -N my_job

cd $PBS_O_WORKDIR
aprun -n 96 -N 12 ./my_executable

- N = number of tasks per node
Might do this to get more memory / task
Note that mppwidth has changed accordingly
Sample Edison Batch Script - MPI

#PBS -q debug
#PBS -l mppwidth=48
#PBS -l walltime=00:10:00
#PBS -N my_job

cd $PBS_O_WORKDIR
aprun -n 96 -j 2 ./my_executable

-j = Turn on hyperthreading
Hybrid OpenMP/MPI

```bash
#PBS -q regular
#PBS -l mppwidth=96
#PBS -l walltime=00:10:00
#PBS -N my_job

cd $PBS_O_WORKDIR
export OMP_NUM_THREADS=6
aprun -n 16 -d 6 -N 4 -S 2 ./hybrid.x
```

A more complex example for mixing MPI and OpenMP:
- 16 tasks (\(-n\)), 4 on each node (\(-N\)), 6 OpenMP threads per task (\(-d\)),
  assign 2 tasks to each NUMA node (\(-s\))

Many more examples on www.nersc.gov
Carver - IBM iDataPlex

- 9,984 compute cores
- 1,202 compute nodes
- 2 quad-core Intel Nehalem 2.67 GHz processors per node
- 8 processor cores per node
- 24 GB of memory per node (48 GB on 80 "fat" nodes)
- Interconnect: InfiniBand 4X QDR

- NERSC global /scratch directory quota of 20 TB
- Full Linux operating system
- PGI, GNU, Intel compilers
Sample Carver Batch Script - MPI

```
#PBS -q debug
#PBS -l nodes=16:ppn=8
#PBS -l walltime=00:10:00
#PBS -N my_job

cd $PBS_O_WORKDIR
mpirun -n 128 ./myexecutable
```

Number of MPI tasks (similar to aprun’s –n argument)  
Distribution across compute cores primarily controlled via the  
ppn value on the PBS –l directive (other ways possible)
Interactive Parallel Jobs

• You can run small parallel jobs interactively for up to 30 minutes (ex. is for Edison)

```
login% qsub -I -l mppwidth=48
[wait for job to start]
mom% cd $PBS_O_WORKDIR
mom% aprun -n 48 ./mycode.x
```
Serial Jobs

• Both Carver and the Crays have a special queue for running serial jobs
  – A single process running on a single core
  – Each serial node can run up to 12 jobs from different users on Carver and 24 jobs on Hopper or Edison

```bash
#PBS -q serial
#PBS -l walltime=00:10:00
#PBS -N my_job

cd $PBS_O_WORKDIR
./myexecutable
```
Monitoring Your Job

• Once your job is submitted, it will start when resources are available

• You can monitor it with:
  – qstat -a
  – qstat -u username
  – showq
  – qs
  – NERSC web site
    https://www.nersc.gov/users/live-status/global-queue-look/
    https://www.nersc.gov/users/job-logs-and-analytics/completed-jobs/
  – MyNERSC
    https://my.nersc.gov
I qsub’d my job, but it’s not running!

- You are not alone!
How do we handle the demand?

That’s what the batch queue is for!

- Your jobs will wait until the resources are available for them to run.
- Your job’s place in the queue is a mix of time and priority, so line jumping is allowed, but it may cost more.
## Job Limits

There are per user, per machine job limits. Here are the limits on Edison as of October 16, 2014.

Specify these queues with:

```bash
#PBS -q queue_name
```

Not these!

<table>
<thead>
<tr>
<th>Submit Queue</th>
<th>Execution Queue</th>
<th>Nodes</th>
<th>Physical Cores</th>
<th>Max Wallclock (hours)</th>
<th>Relative Priority</th>
<th>Run Limit</th>
<th>Queued Limit</th>
<th>Queue Charge Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>debug</td>
<td>debug</td>
<td>1-512</td>
<td>1-12,288</td>
<td>30 mins</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>ccm_int(^1)</td>
<td>ccm_int</td>
<td>1-512</td>
<td>1-12,288</td>
<td>30 mins</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>regular</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>reg_small</td>
<td>1-682</td>
<td>1-16,368</td>
<td>48 hrs</td>
<td>3</td>
<td>24</td>
<td>24</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>reg_med</td>
<td>683-2048</td>
<td>16,369-49,152</td>
<td>36 hrs</td>
<td>3</td>
<td>8</td>
<td>8</td>
<td>0.6</td>
</tr>
<tr>
<td></td>
<td>reg_big</td>
<td>2049-4096</td>
<td>49,153-98,304</td>
<td>36 hrs</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0.6</td>
</tr>
<tr>
<td></td>
<td>reg_xbig</td>
<td>4097-5462</td>
<td>98,305-131,088</td>
<td>12 hrs</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0.6</td>
</tr>
<tr>
<td>ccm_queue</td>
<td>ccm_queue</td>
<td>1-682</td>
<td>1-16,368</td>
<td>48 hrs</td>
<td>3</td>
<td>16</td>
<td>16</td>
<td>1</td>
</tr>
<tr>
<td>premium</td>
<td>premium</td>
<td>1-2048</td>
<td>1-49,152</td>
<td>36</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>low</td>
<td>low</td>
<td>1-682</td>
<td>1-16,368</td>
<td>24</td>
<td>4</td>
<td>8</td>
<td>8</td>
<td>0.5</td>
</tr>
<tr>
<td>killable(^2)</td>
<td>killable</td>
<td>1-682</td>
<td>1-16,368</td>
<td>48 hrs</td>
<td>3</td>
<td>8</td>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>serial(^3)</td>
<td>serial</td>
<td>1</td>
<td>1</td>
<td>48 hrs</td>
<td>-</td>
<td>50</td>
<td>50</td>
<td>1/24</td>
</tr>
</tbody>
</table>
Tips for jobs

• Submit shorter jobs, they are easier to schedule
  – Checkpoint if possible to break up long jobs
  – Short jobs can take advantage of ‘backfill’ opportunities
  – Run short jobs just before maintenance

• Very important: make sure the wall clock time you request is accurate
  – As noted above, shorter jobs are easier to schedule
  – Many users unnecessarily enter the largest wall clock time possible as a default
How Your Jobs Are Charged

• Your repository is charged for each node your job was allocated for the entire duration of your job.
  – The minimum allocatable unit is a node (except for the serial queues). Hopper and Edison have 24 cores / node, so your minimum charge is 24*walltime.

\[
\text{MPP hours} = (\# \text{ nodes}) \times (\# \text{ cores / node}) \times (\text{walltime}) \times (\text{QCF}) \times (\text{MCF})
\]

  – Example: 96 Edison cores for 1 hour in regular queue
    \[
    \text{MPP hours} = (4) \times (24) \times (1 \text{ hour}) \times (1) \times (2) = 192 \text{ MPP hours}
    \]
  – Serial jobs are charged with: (walltime) \times (MCF)

• If you have access to multiple repos, pick which one to charge in your batch script

  \#PBS –A repo_name
Charge Factors & Discounts

• Each machine has a “machine charge factor” (MCF) that multiplies the “raw hours” used
  – Edison MCF = 2.0
  – Hopper MCF = 1.0
  – Carver MCF = 1.5

• Each queue has a “queue charge factor” (QCF) and corresponding relative scheduling priorities
  – Premium QCF = 2.0
  – Low QCF = 0.5
  – Regular (and everything else) QCF = 1.0 (Hopper: 0.8)

• On Edison:
  – Jobs requesting more than 682 nodes (reg_med, reg_big, reg_xbig queues) get a 40% discount (QCF = 0.6)
NERSC Web pages

Hopper:  
http://www.nersc.gov/users/computational-systems/hopper/running-jobs/

Edison:  
http://www.nersc.gov/users/computational-systems/edison/running-jobs/

Carver:  
http://www.nersc.gov/users/computational-systems/carver/running-jobs/

Contact NERSC Consulting:
- Toll-free 800-666-3772
- 510-486-8611, #3
- Email consult@nersc.gov.
Thank You
## Average Queue Wait Time

| Hours Requested | Nodes | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48+ |
| 1.5            | 5.5  | 6.5  | 7.5  | 8.5  | 9.5  | 10.5 | 11.5 | 12.5 | 13.5 | 14.5 | 15.5 | 16.5 | 17.6 | 18.6 | 19.6 | 20.6 | 21.6 | 22.6 | 23.6 | 24.6 | 25.6 | 26.6 | 27.6 | 28.6 | 29.6 | 30.6 | 31.6 | 32.6 | 33.6 | 34.6 | 35.6 | 36.6 | 37.6 | 38.6 | 39.6 | 40.6 | 41.6 | 42.6 | 43.6 | 44.6 | 45.6 | 46.6 | 47.6 | 48.6 | 49.6+ |
| 3.0            | 7.0  | 8.0  | 9.0  | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.6 | 17.6 | 18.6 | 19.6 | 20.6 | 21.6 | 22.6 | 23.6 | 24.6 | 25.6 | 26.6 | 27.6 | 28.6 | 29.6 | 30.6 | 31.6 | 32.6 | 33.6 | 34.6 | 35.6 | 36.6 | 37.6 | 38.6 | 39.6 | 40.6 | 41.6 | 42.6 | 43.6 | 44.6 | 45.6 | 46.6 | 47.6 | 48.6 | 49.6 | 50.6+ |
| 6.0            | 12.0 | 14.0 | 16.0 | 18.0 | 20.0 | 22.0 | 24.0 | 26.0 | 28.0 | 29.6 | 30.6 | 31.6 | 32.6 | 33.6 | 34.6 | 35.6 | 36.6 | 37.6 | 38.6 | 39.6 | 40.6 | 41.6 | 42.6 | 43.6 | 44.6 | 45.6 | 46.6 | 47.6 | 48.6 | 49.6 | 50.6 | 51.6 | 52.6 | 53.6 | 54.6 | 55.6 | 56.6 | 57.6 | 58.6 | 59.6 | 60.6 | 61.6+ |

### Note
- The table above illustrates the average queue wait time in hours based on the requested hours. Each row represents a specific number of requested hours with the corresponding wait times below. The wait times are given in a specific format, indicating the average time a job would wait in the queue. For instance, a request of 1.5 hours waits an average of 5.5 hours, indicating a significant delay in job execution due to the queue's dynamics.