Introduction to LONI Computing

Resources

Le Yan

HPC Consultant
User Services
Outline

- Hardware overview
- Software environment
- Job management
Outline

- Hardware overview
- Software environment
- Job management
Architectures of LONI Clusters

- Two platforms
  - Linux clusters
    - Vendor: Dell
    - Operating System: Linux (Red hat)
    - Processor: Intel
  - AIX clusters
    - Vendor: IBM
    - Operating System: AIX
    - Processor: IBM
Current deployment status - Dell Linux clusters

<table>
<thead>
<tr>
<th>Name</th>
<th>Peak TeraFLOPS/s</th>
<th>Location</th>
<th>Status</th>
<th>Login</th>
</tr>
</thead>
<tbody>
<tr>
<td>Queen Bee</td>
<td>50.7</td>
<td>ISB</td>
<td>Available</td>
<td>LONI</td>
</tr>
<tr>
<td>Eric</td>
<td>4.7</td>
<td>LSU</td>
<td>Available</td>
<td>LONI</td>
</tr>
<tr>
<td>Oliver</td>
<td>4.7</td>
<td>ULL</td>
<td>Available</td>
<td>LONI</td>
</tr>
<tr>
<td>Louie</td>
<td>4.7</td>
<td>Tulane</td>
<td>Available</td>
<td>LONI</td>
</tr>
<tr>
<td>Poseidon</td>
<td>4.7</td>
<td>UNO</td>
<td>Available</td>
<td>LONI</td>
</tr>
<tr>
<td>Painter</td>
<td>4.7</td>
<td>LaTech</td>
<td>Available</td>
<td>LONI</td>
</tr>
</tbody>
</table>

LONI High Performance Computing Workshop – University of New Orleans
March 29-30, 2010
Current deployment status - IBM AIX clusters

<table>
<thead>
<tr>
<th>Name</th>
<th>Peak TeraFLOPS/s</th>
<th>Location</th>
<th>Status</th>
<th>Login</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bluedawg</td>
<td>0.85</td>
<td>LaTech</td>
<td>Available</td>
<td>LONI</td>
</tr>
<tr>
<td>Ducky</td>
<td>0.85</td>
<td>Tulane</td>
<td>Available</td>
<td>LONI</td>
</tr>
<tr>
<td>Zeke</td>
<td>0.85</td>
<td>ULL</td>
<td>Available</td>
<td>LONI</td>
</tr>
<tr>
<td>Neptune</td>
<td>0.85</td>
<td>UNO</td>
<td>Available</td>
<td>LONI</td>
</tr>
<tr>
<td>Lacumba</td>
<td>0.85</td>
<td>Southern</td>
<td>Available</td>
<td>LONI</td>
</tr>
</tbody>
</table>
Cluster Architecture

- A cluster is a group of computers (nodes) that works together closely
- Most have high speed interconnect
- Type of nodes
  - Head node
  - Compute node
Hardware Specification

- **Queen Bee**
  - 668 nodes, each has 8 Intel Xeon cores @ 2.33 GHz, 8 GB RAM
  - 192 TB storage

- **Other LONI Linux clusters**
  - 128 nodes: each has 4 Intel Xeons cores @ 2.33 GHz, 4 GB RAM
  - 9 TB storage

- **LONI AIX clusters**
  - 14 nodes: each has 8 IBM Power5 processors @ 1.9 GHz, 16 GB RAM
  - 280 GB storage
Hardware Specification

- **Queen Bee**
  - **668 nodes**, each has 8 Intel Xeon cores @ 2.33 GHz, 8 GB RAM
  - 192 TB storage
- **Other LONI Linux clusters**
  - **128 nodes**: each has 4 Intel Xeons cores @ 2.33 GHz, 4 GB RAM
  - 9 TB storage
- **LONI AIX clusters**
  - **14 nodes**: each has 8 IBM Power5 processors @ 1.9 GHz, **16 GB RAM**
  - 280 GB storage
How Much Memory Your Program Can Use

- The amount of installed memory less the amount that is used by the operating system and other utilities
- Max amount per node
  - Linux clusters: ~6 GB for Queen Bee, ~3 GB for others
  - AIX clusters: ~13 GB
Choose The Correct Architecture/Platform

- There are numerous different architectures in the HPC world
- You need to choose the correct one when installing or using software
  - Linux clusters
    - EM64T, AMD64, X86_64
  - AIX clusters:
    - PowerPC, Power5
Outline

- Hardware overview
- Software environment
- Job management
Accessing LONI Clusters

- Host name: `<cluster name>.loni.org`
  - Queen Bee: `qb.loni.org`
- Use ssh to connect
  - *nix and Mac: type "ssh `<host name>`" in a terminal
  - Windows: use Putty
- Only accessible via Internet 2 at the moment
- The default Login shell is bash
  - Supported shells: bash, tcsh, ksh, csh & sh
  - Change the login shell at the profile page
    - Log in at allocations.loni.org and click on “profile”
Exercise 1

- Log in any cluster of your choice
- Locate the directory `/home/lyan1/traininglab/environment`
  - There are files that you will need for following exercises
  - Copy them to your user space
File Systems

<table>
<thead>
<tr>
<th></th>
<th>Distributed file system</th>
<th>Throughput</th>
<th>File life time</th>
<th>Best used for</th>
</tr>
</thead>
<tbody>
<tr>
<td>Home</td>
<td>Yes</td>
<td>Low</td>
<td>Unlimited</td>
<td>Code in development, compiled executables</td>
</tr>
<tr>
<td>Work</td>
<td>Yes</td>
<td>High</td>
<td>30 days</td>
<td>Job input/output</td>
</tr>
<tr>
<td>Local Scratch</td>
<td>No</td>
<td></td>
<td>Job duration</td>
<td>Temporary files needed by running jobs</td>
</tr>
</tbody>
</table>

- Never let you job write output to your home directory
- Nothing is backed up
- Do not write temporary files to /tmp
  - Write to the local scratch or work space
- The work space is not for long-term storage
  - Files are purged periodically
- Use “rmpurge” to delete large amount of files
- Don't store over 10,000 files in one directory
Disk Quota

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Home Access point</th>
<th>Home Quota</th>
<th>Work Access point</th>
<th>Work Quota</th>
<th>Local scratch Access point</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONI Linux</td>
<td>/home/$USER</td>
<td>5 GB</td>
<td>/work/$USER</td>
<td>100 GB</td>
<td>/var/scratch</td>
</tr>
<tr>
<td>LONI AIX</td>
<td>/home/$USER</td>
<td>500 MB</td>
<td>/work/default/$USER</td>
<td>20 GB</td>
<td>/scratch/local</td>
</tr>
</tbody>
</table>

- No quota is enforced on the work space on Queen Bee
- On Linux clusters, the work directory is created within an hour after the first login
- Check current disk usage
  - **Linux**: `showquota`
  - **AIX**: `quota`
Transferring Files

- From Windows machines to clusters
  - Winscp
  - Filezilla

- Between clusters
  - `scp`: easiest for small transfers
  - `bbcp`: faster alternative to `scp`
  - `rsync`: synchronizes entire directory trees
  - `gridftp`: allows for parallel streams

https://docs.loni.org/wiki/Transferring_Files
Setting up Environment Variables

• Environment variables
  • **PATH**: where to look for executables
  • **LD_LIBRARY_PATH**: where to look for shared libraries
  • Other custom environment variables needed by various software packages

• **SOFTENV** is a software that can be used to set these environment variables
  • Manages a database of installed software packages
  • More convenient than using .bashrc or .cshrc
Listing All Packages

• Command “softenv” lists all packages that are managed by SOFTENV

[lryan1@tezpur2 ~]$ softenv
...
These are the macros available:
* @default
* @globus-4.0
* @intel-compilers

These are the keywords explicitly available:
+Mesa-6.4.2
+R-2.8.0-gcc-3.4.6
+ansys-1sdyna-11.0

Softenv key

---
No description yet for Mesa-6.4.2.
application: 'R', version 2.8.0
application: 'ANSYS LS-DYNA', version: 11.0
ANSYS LS-DYNA is a premier software package for explicit nonlinear structural simulation with finite element pre- and post-processor. docs =>
http://www1.ansys.com/customer/
Searching A Specific Package

- Use “–k” option with “softenv”

[lyan1@tezpur2 ~] $ softenv –k fftw
SoftEnv version 1.6.4

... Search Regexp: fftw

These are the macros available:

+fftw-3.1.2-gnu  application: FFTW, version 3.1.2, binded with GNU compiler.
Setting up Environment via Softenv – one time change

- Set up the environment variables to use a certain software package in the current login session only
  - Add a package: `soft add <key>`
  - Remove a package: `soft delete <key>`

```
[lyan1@tezpur2 ~]$ which gcc
/usr/local/compilers/GNU/gcc-4.2.0/bin/gcc
[lyan1@tezpur2 ~]$ soft add +gcc-4.3.0
[lyan1@tezpur2 ~]$ which gcc
/usr/local/compilers/GNU/gcc-4.3.0/bin/gcc
[lyan1@tezpur2 ~]$ soft delete +gcc-4.3.0
[lyan1@tezpur2 ~]$ which gcc
/usr/local/compilers/GNU/gcc-4.2.0/bin/gcc
```
Setting up Environment via Softenv – permanent change

- Set up the environment variables to use a certain software package
  - First add the key to $HOME/.soft
  - Then execute `resoft` at the command line
  - The environment will be the same next time you log in

```
[lyan1@tezpur2 ~]$ cat .soft
#
# This is the .soft file.
...
+matlab-r2007b
@default
[lyan1@tezpur2 ~]$ resoft
```
Querying a Softenv key

- **Command** “soft-dbq” shows which variables are set by a SOFTENV key

```
[lyan1@tezpur2 ~]$ soft-dbq +gcc-4.3.0
This is all the information associated with the key or macro +gcc-4.3.0.

-----------------------------------------------
Name: +gcc-4.3.0
Description: GNU gcc compiler, version 4.3.0
Flags: none
Groups: none
Exists on: Linux

-----------------------------------------------
On the Linux architecture,
the following will be done to the environment:
The following environment changes will be made:
  LD_LIBRARY_PATH = ${LD_LIBRARY_PATH}:/usr/local/compilers/GNU/gcc-4.3.0/lib64
  PATH = ${PATH}:/usr/local/compilers/GNU/gcc-4.3.0/bin
```

---

LONI High Performance Computing Workshop – University of New Orleans
March 29-30, 2010
Exercise 2: Use Softenv

- Find the key for VISIT (a visualization package)
- Check what variables are set through the key
- Set up your environment to use VISIT
- Check if the variables are correctly set by "which visit"
Exercise 2: Use Softenv

- Find the key for VISIT (a visualization package)
  - Use `softenv -k visit`

- Check what variables are set through the key
  - Use `soft-dbq +visit-1.12.1-gcc-4.3.2`

- Set up your environment to use VISIT
  - Use `soft add +visit-1.12.1-gcc-4.3.2`
  - Or add “+visit-1.12.1-gcc-4.3.2” to your .soft file and resoft

- Check if the variables are correctly set by “which visit”
  - The output should be the path to the executable `visit`
# Compilers

<table>
<thead>
<tr>
<th>Language</th>
<th>Linux clusters</th>
<th>AIX clusters</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Intel</td>
<td>GNU</td>
</tr>
<tr>
<td>Fortran</td>
<td>ifort</td>
<td>g77</td>
</tr>
<tr>
<td>C</td>
<td>icc</td>
<td>gcc</td>
</tr>
<tr>
<td>C++</td>
<td>icpc</td>
<td>g++</td>
</tr>
</tbody>
</table>

- **Usage:** `<compiler> <options> <your_code>`
  - **Example:** `icc -O3 -o myexec mycode.c`

- **Some compilers options are architecture specific**
  - Linux: EM64T, AMD64 or X86_64
  - AIX: power5 or powerpc
MPI Compilers (1)

<table>
<thead>
<tr>
<th>Language</th>
<th>Linux clusters</th>
<th>AIX clusters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fortran</td>
<td>mpif77, mpif90</td>
<td>mpixlf, mpixlf_r, mpixlf90, mpixlf90_r</td>
</tr>
<tr>
<td>C</td>
<td>mpicc</td>
<td>mpcc, mpcc_r</td>
</tr>
<tr>
<td>C++</td>
<td>mpiCC</td>
<td>mpCC, mpCC_r</td>
</tr>
</tbody>
</table>

- Usage: similar to what we have seen
  - Example: `mpif90 -O2 -o myexec mycode.f90`

- On Linux clusters
  - Only one compiler for each language
  - There is no intel_mpicc or pg_mpicc
MPI Compilers (2)

- These MPI compilers are actually **wrappers**
  - They still use the compilers we've seen on the previous slide
    - Intel, PGI or GNU
  - They take care of everything we need to build MPI codes
    - Head files, libraries etc.
  - What they actually do can be reveal by the `–show` option

```
[lyan1@tezpur2 ~]$ mpicc –show
icc –DUSE_STDARG –DHAVE_STDLIB_H=1 –DHAVE_STRING_H=1 –DHAVE_UNISTD_H=1
  –DHAVE_STDARG_H=1 –DUSE_STDARG=1 –DMALLOC_RET VOID=1
  –L/usr/local/packages/mvapich-1.0-intel10.1/lib –lmpich
  –L/usr/local/ofed/lib64 –Wl,-rpath=/usr/local/ofed/lib64 –libverbs
  –libumad –lpthread –lpthread –lrt
```
MPI Compilers (3)

There are many different versions of MPI compilers on Linux clusters
Each of them is built around a specific compiler
  • Intel, PGI or GNU

It is extremely important to compile and run your code with the same version!!!
Application Packages

- **Installed under** `/usr/local/packages`
- **Most of them are managed by SOFTENV**
  - Numeric and utility libraries
    - FFTW, HDF5, NetCDF, PETSc, MKL
  - Computational chemistry
    - Amber, Gaussian, CPMD, NWChem, NAMD, Gromacs
  - Profiling/debugging tools
    - TAU, Totalview
  - ...

<http://www.hpc.lsu.edu>
Exercise 3: Compile a code

• Serial code
  • Copy hello.f90
    from /home/lyan1/traininglab/environment
  • Compile it with a compiler of your choice
  • Run the executable from the command line

• MPI code
  • Copy hello_mpi.f90
    from /home/lyan1/traininglab/environment
  • Compile it with a serial compiler and see what happens
  • Compile it with an MPI compiler
Exercise 3: Compile a code

- **Linux**
  - cp /home/lyan1/traininglab/environment/* .f90
  - icc -o hello_ser hello.f90
  - ./hello_ser
  - mpif90 -o hello hello_mpi.f90
- **AIX**
  - cp /home/lyan1/traininglab/environment/* .f90
  - xlf90_r -o hello_ser hello.f90
  - ./hello_ser
  - mpxlf90_r -o hello hello_mpi.f90
  - We will run it later
Outline

- Hardware overview
- Software environment
- Job management
Cluster Environment

- Multiple compute nodes
- Multiple users
- Each user may have multiple jobs running simultaneously
Batch Queuing System

- A software that manages resources (CPU time, memory etc.) and schedules job execution
  - Linux clusters: Portable Batch System (PBS)
  - AIX clusters: Loadleveler
- What is a job
  - A user's request to use a certain amount of resources for a certain amount of time
- The batch queuing system determines
  - The order jobs are executed
  - On which node(s) jobs are executed
Job Scheduling

- Map jobs onto the node-time space
  - Assuming CPU time is the only resource
- Need to
  - Honor the order in which jobs are received
  - Maximize resource utilization
Backfilling

- A strategy to improve utilization
  - Allow a job to jump ahead of others when there are enough idle nodes
  - Must not affect the estimated start time of the job with the highest priority
- Enabled on all LONI clusters
How Much Time Should I Ask for?

- Ask for an amount of time that is
  - Long enough for your job to complete
  - As short as possible to increase the chance of backfilling
Job Queues

- There are more than one job queue
- Each job queue differs in
  - Type of jobs (single processor vs. parallel)
  - Number of available nodes
  - Max run time
  - Max running jobs per user
  - ...
- The main purpose is to maximize utilization
# Queue Characteristics – Queen Bee

<table>
<thead>
<tr>
<th>Queue</th>
<th>Max Runtime</th>
<th>Total number of available nodes</th>
<th>Max running jobs per user</th>
<th>Max nodes per job</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workq</td>
<td>2 days</td>
<td>530</td>
<td>8</td>
<td>128</td>
<td>Unpreemptable (default)</td>
</tr>
<tr>
<td>Checkpt</td>
<td></td>
<td>668</td>
<td></td>
<td>256</td>
<td>Preemptable jobs</td>
</tr>
<tr>
<td>Preempt</td>
<td></td>
<td>668</td>
<td>NA</td>
<td></td>
<td>Require permission</td>
</tr>
<tr>
<td>Priority</td>
<td></td>
<td>668</td>
<td>NA</td>
<td></td>
<td>Require permission</td>
</tr>
</tbody>
</table>
## Queue Characteristics – Other LONI Linux Clusters

<table>
<thead>
<tr>
<th>Queue</th>
<th>Max Runtime</th>
<th>Total number of available nodes</th>
<th>Max running jobs per user</th>
<th>Max nodes per job</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single</td>
<td>14 days</td>
<td>16</td>
<td>64</td>
<td>1</td>
<td>Single processor jobs</td>
</tr>
<tr>
<td>Workq</td>
<td>3 days</td>
<td>96</td>
<td>8</td>
<td>40</td>
<td>Unpreemptable (default)</td>
</tr>
<tr>
<td>Checkpt</td>
<td>3 days</td>
<td>128</td>
<td>8</td>
<td>64</td>
<td>Preemptable jobs</td>
</tr>
<tr>
<td>Preempt</td>
<td>3 days</td>
<td>64</td>
<td>NA</td>
<td>NA</td>
<td>Require permission</td>
</tr>
<tr>
<td>Priority</td>
<td>3 days</td>
<td>64</td>
<td>NA</td>
<td>NA</td>
<td>Require permission</td>
</tr>
</tbody>
</table>
## Queue Characteristics – LONI AIX Clusters

<table>
<thead>
<tr>
<th>Queue</th>
<th>Max Runtime</th>
<th>Total number of available nodes</th>
<th>Max running jobs per user</th>
<th>Max nodes per job</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single</td>
<td>14 days</td>
<td>1</td>
<td>8</td>
<td>1</td>
<td>Single processor jobs</td>
</tr>
<tr>
<td>Workq</td>
<td>5 days</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>Unpreemptable (default)</td>
</tr>
<tr>
<td>Checkpt</td>
<td>5 days</td>
<td>14</td>
<td>NA</td>
<td>14</td>
<td>Preemptable jobs</td>
</tr>
<tr>
<td>Preempt</td>
<td>6</td>
<td>NA</td>
<td>NA</td>
<td></td>
<td>Require permission</td>
</tr>
<tr>
<td>Priority</td>
<td>6</td>
<td>NA</td>
<td>NA</td>
<td></td>
<td>Require permission</td>
</tr>
</tbody>
</table>
Basic Commands

- Queue querying
  - Check how busy the cluster is
- Job submission
- Job monitoring
  - Check job status (estimated start time, remaining run time etc.)
- Job manipulation
  - Cancel/hold jobs
Queue Querying – Linux Clusters

- **Command:** `qfree`
  - Show the number of free, busy and queued nodes
- **Command:** `qfreeloni`
  - Equivalent to run `qfree` on all LONI Linux clusters

```
[lyan1@louie2 ~]$ qfree
PBS total nodes: 128, free: 81, busy: 44, down: 3, use: 34%
PBS checkpt nodes: 128, free: 81, busy: 28
PBS workq nodes: 32, free: 16, busy: 16
```
Queue Querying – AIX Clusters

- **Command**: `llclass`

```bash
lyan1012f1n03$ llclass
Name        MaxJobCPU  MaxProcCPU  Free   Max  Description
            d+hh:mm:ss  d+hh:mm:ss  Slots  Slots
-----------------  -----------------  ------  ------  --------------------------
interactive     undefined  undefined   8     8     Interactive Parallel jobs running on interactive node
single          unlimited  unlimited   4     8     One node queue (14 days) for serial and up to 8 processor parallel jobs
workq           unlimited  unlimited   51    56    Default queue (5 days), up to 56 processors
priority        unlimited  unlimited   40    40    priority queue reserved for on-demand jobs (5 days), up to 48 processors
preempt         unlimited  unlimited   40    40    preemption queue reserved for on-demand jobs (5 days), up to 48 processors
checkpt         unlimited  unlimited   91    96    queue for checkpointing jobs (5 days), up to 104 processors, Job running on this queue can be preempted for on-demand job
```

---

`llclass` command is used to query the status of queues in an AIX cluster. It shows the name of the queue, maximum job CPU time, maximum processor CPU time, free and maximum slots, and a description of the queue's purpose and limitations. For example, the `interactive` queue allows parallel jobs to run on an interactive node with a maximum of 8 slots, while the `single` queue allows one node queue (14 days) for serial and up to 8 processor parallel jobs with a maximum of 8 slots. The `workq` queue is a default queue (5 days) that allows up to 56 processors with 51 free slots and 56 maximum slots. The `priority` queue is reserved for on-demand jobs (5 days) with a maximum of 40 processors, allowing up to 40 slots. The `preempt` queue is similar to the `priority` queue but is for preemption with a maximum of 40 processors and 40 slots. The `checkpt` queue is for checkpointing jobs (5 days) and allows up to 104 processors, with 91 free slots and 96 maximum slots. Jobs running on this queue can be preempted for on-demand jobs.
Checking Loads on All LONI Clusters

- Check Loads on all LONI clusters at docs.loni.org
- Updated every 15 minutes

### Dell Linux Clusters

<table>
<thead>
<tr>
<th>System Name</th>
<th>Nodes</th>
<th>SMP Size</th>
<th>Total CPUs</th>
<th>Memory/Node</th>
<th>TFLOPS</th>
<th>Work Disk</th>
<th>Location</th>
<th>Load</th>
<th>Running jobs</th>
<th>Queued jobs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Queen Bee</td>
<td>680</td>
<td>8</td>
<td>5440</td>
<td>8 GB</td>
<td>50.7</td>
<td>58 TB</td>
<td>LSU</td>
<td>0</td>
<td>422</td>
<td></td>
</tr>
<tr>
<td>Enc</td>
<td>128</td>
<td>4</td>
<td>512</td>
<td>4 GB</td>
<td>4.772</td>
<td>9 TB</td>
<td>LSU</td>
<td>70</td>
<td>111</td>
<td></td>
</tr>
<tr>
<td>Oliver</td>
<td>128</td>
<td>4</td>
<td>512</td>
<td>4 GB</td>
<td>4.772</td>
<td>9 TB</td>
<td>ULL</td>
<td>16</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>Louie</td>
<td>128</td>
<td>4</td>
<td>512</td>
<td>4 GB</td>
<td>4.772</td>
<td>9 TB</td>
<td>Tulane</td>
<td>27</td>
<td>56</td>
<td></td>
</tr>
<tr>
<td>Poseidon</td>
<td>128</td>
<td>4</td>
<td>512</td>
<td>4 GB</td>
<td>4.772</td>
<td>9 TB</td>
<td>UNO</td>
<td>17</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Painter</td>
<td>128</td>
<td>4</td>
<td>512</td>
<td>4 GB</td>
<td>4.772</td>
<td>9 TB</td>
<td>LaTech</td>
<td>23</td>
<td>28</td>
<td></td>
</tr>
</tbody>
</table>

### IBM P5 Clusters

<table>
<thead>
<tr>
<th>System Name</th>
<th>Nodes</th>
<th>SMP Size</th>
<th>Total CPUs</th>
<th>Memory/Node</th>
<th>TFLOPS</th>
<th>Work Disk</th>
<th>Location</th>
<th>Load</th>
<th>Running jobs</th>
<th>Queued jobs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bluedawg</td>
<td>14</td>
<td>8</td>
<td>104</td>
<td>16 GB</td>
<td>0.85L</td>
<td>270 GB</td>
<td>LaTech</td>
<td>16</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Ducky</td>
<td>14</td>
<td>8</td>
<td>104</td>
<td>16 GB</td>
<td>0.85L</td>
<td>270 GB</td>
<td>Tulane</td>
<td>7</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Zeke</td>
<td>14</td>
<td>8</td>
<td>104</td>
<td>16 GB</td>
<td>0.85L</td>
<td>270 GB</td>
<td>ULL</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Neptune</td>
<td>14</td>
<td>8</td>
<td>104</td>
<td>16 GB</td>
<td>0.85L</td>
<td>270 GB</td>
<td>UNO</td>
<td>9</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>LaCumba</td>
<td>14</td>
<td>8</td>
<td>104</td>
<td>16 GB</td>
<td>0.85L</td>
<td>270 GB</td>
<td>SU</td>
<td>8</td>
<td>8</td>
<td></td>
</tr>
</tbody>
</table>
Job Types

• Interactive job
  • Set up an interactive environment on compute nodes for users
    • Advantage: can run programs interactively
    • Disadvantage: must be present when the job starts
  • Purpose: testing and debugging
  • Disabled on AIX clusters because of limited number of nodes

• Batch job
  • Executed without user intervention using a job script
    • Advantage: the system takes care of everything
    • Disadvantage: can only execute one sequence of commands which cannot changed after submission
  • Purpose: production run
Submitting Jobs – Linux Clusters

- **Interactive job**
  - `qsub -I -V -l walltime=<hh:mm:ss>,nodes=<# of nodes>:ppn=4 -A <your allocation> -q <queue name>`

- **Batch job**
  - `qsub <job script>`

- **ppn must be either 4 (all Linux clusters except Queen Bee) or 8 (Queen Bee) except for serial jobs**
PBS Job Script – Parallel Jobs

#!/bin/bash
#PBS -l nodes=4:ppn=4
#PBS -l walltime=24:00:00
#PBS -N myjob
#PBS -o <file name>
#PBS -e <file name>
#PBS -q checkpt
#PBS -A <loni_allocation>
#PBS -m e
#PBS -M <email address>

<shell commands>
mpirun -machinefile $PBS_NODEFILE -np 16 <path_to_executable> <options>
<shell commands>

Number of nodes and processor
Maximum wall time
Job name
File name for standard output
File name for standard error
Queue name
Allocation name
Send mail when job ends
Send mail to this address
PBS Job Script – Serial Jobs

```bash
#!/bin/bash
#PBS -l nodes=1:ppn=1
#PBS -l walltime=24:00:00
#PBS -N myjob
#PBS -o <file name>
#PBS -e <file name>
#PBS -q single
#PBS -A <loni_allocation>
#PBS -m e
#PBS -M <email address>

<shell commands>
<path_to_executable> <options>
<shell commands>
```

- Number of nodes and processor
- Maximum wall time
- Job name
- File name for standard output
- File name for standard error
- The only queue that accepts serial jobs
- Allocation name
- Send mail when job ends
- Send mail to this address
Submitting Jobs - AIX Clusters

- Batch job

- `llsubmit <job script>`

```
#!/bin/sh
#@ job_type = parallel
#@ output = /work/default/username/$(jobid).out
#@ error = /work/default/username/$(jobid).err
#@ notify_user = youremail@domain
#@ notification = error
#@ class = checkpt
#@ wall_clock_limit = 24:00:00
#@ node_usage = shared
#@ node = 2,2
#@ total_tasks = 16
#@ initialdir = /work/default/username
#@ environment = COPY_ALL
#@ queue

<shell commands>
poe <path_to_executable> <options>
<shell commands>
```
Loadleveler Job Script – Serial Jobs

#!/bin/sh
#@ job_type = serial
#@ output = /work/default/username/$(jobid).out
#@ error = /work/default/username/$(jobid).err
#@ notify_user = youremail@domain
#@ notification = error
#@ class = checkpoint
#@ wall_clock_limit = 24:00:00
#@ initialdir = /work/default/username
#@ environment = COPY_ALL
#@ queue

<shell commands>
<path_to_executable> <options>
<shell commands>
Job Monitoring – Linux Clusters

- **Command:** `showstart <job_id>
  - Check when a job is estimated to start
- Things that can change the estimated start time
  - Higher priority job gets submitted
  - Other jobs terminate earlier than the system expects
  - The system has trouble starting your job
Job Monitoring – Linux Clusters cont'd

- **Command**: `qstat <options> <job_id>`
  - Show information on job status
  - All jobs are displayed if `<job_id>` is omitted
  - Show jobs submitted by a specific user: `qstat -u <username>`
  - Display in the alternative format: `qstat -a <job_id>`

- **Command**: `qshow <job_id>`
  - Show information on a running job
    - On which node(s) the job is running
    - CPU load
    - Memory usage
Job Monitoring – AIX Clusters

- **Command:** `llq <options> <job_id>
  - All jobs are displayed if `<job_id>` is omitted
  - Display detailed information: `llq -l <job_id>`
  - Check the estimated start time: `llq -s <job_id>`
  - Show jobs from a specific user: `llq -u <username>`

```
lyan1@l2f1n03$ llq

<table>
<thead>
<tr>
<th>Id</th>
<th>Owner</th>
<th>Submitted</th>
<th>ST</th>
<th>PRI</th>
<th>Class</th>
<th>Running On</th>
</tr>
</thead>
<tbody>
<tr>
<td>l2f1n03.3697.0</td>
<td>collin</td>
<td>1/22 16:59 R 50 single</td>
<td>l2f1n14</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>l2f1n03.3730.0</td>
<td>jheiko</td>
<td>1/28 13:30 R 50 workq</td>
<td>l2f1n10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>l2f1n03.3726.0</td>
<td>collin</td>
<td>1/26 08:21 R 50 single</td>
<td>l2f1n14</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>l2f1n03.3698.0</td>
<td>collin</td>
<td>1/22 17:00 R 50 single</td>
<td>l2f1n14</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>l2f1n03.3727.0</td>
<td>collin</td>
<td>1/26 08:21 R 50 single</td>
<td>l2f1n14</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5 job step(s) in queue, 0 waiting, 0 pending, 5 running, 0 held, 0 preempted
```
Job Manipulation – Linux Clusters

- **Command:** `qdel <job_id>`
  - Cancel a running or queued job
  - May take some time depending on the size of the job

- **Command:** `qhold <job_id>`
  - Put a queued job on hold

- **Command:** `qrls <job_id>`
  - Resume a held job
Job Manipulation – AIX Clusters

- **Command**: `llcancel <job_id>`
  - Cancel a running or queued job
- **Command**: `llhold <job_id>`
  - Put a queued job on hold
- **Command**: `llhold -r <job_id>`
  - Resume a held job
Exercise 4

- Compile the parallel program hello_mpi.f90
  - Located under /home/lyan1/traininglab/environment
  - To compile
    - Linux clusters: mpif90 -o <name of executable> hello_mpi.f90
    - AIX clusters: mpxlf90 -o <name of executable> hello_mpi.f90

- Run it within an interactive job session
  - Submit an interactive job
  - Run on the command line
    - Linux clusters: mpirun -np <# of cpus> <name of executable>
Exercise 5

- Run the same program as a batch job
  - Sample submission scripts can be found under the same directory
    - **Linux clusters**: submit.aix
    - **AIX clusters**: submit.linux