User Environment on LONI and LSU HPC Clusters

Bhupender Thakur
HPC @ LSU
Outline

- Cluster Hardware
- Accessing Software
- Submitting and Monitoring Jobs
**Login nodes** get you access to the cluster. Individual nodes are not accessible.
- Login via ssh
- Node are not meant to run jobs

**Compute nodes** are connected via a network of switches
- QDR switches on SM-II
- Latencies typically few microsecs
- Bandwidth 40Gbps

**Resource managers** give access to compute resource
- PBS/ loadleveler installed
- Run commands qsub, qstat, qdel

---

**General Cluster Architecture**

- **Login node**
- **Switched network**
- **Compute nodes**
- **More Compute nodes**
- **Resource Managers**
- **I/O nodes**

---

2/5/14  HPC User Environment Spring 2014
Available HPC resources

- Hardware resources consist of LONI and LSU HPC cluster systems
- LONI and LSU HPC maintain separate LDAP for authentication. In essence, you need separate accounts
- Both resources are managed centrally by a core team at LSU.
- To get help on either Docs: www.hpc.lsu.edu Help: sys-help@loni.org

High Performance Computing
Louisiana State University

HPC Director CIO CCT Director LONI Director

Home About Us Users Resources Documentation Training Announcements Links

Home About Us Users Resources Documentation Training Announcements Links

Consultation
HPC @ LSU provides consulting services to users of the computational resources. This includes assistance with compiler optimizations, use of specialized libraries, help with MPI issues, guidance on software usage, and more. See the Help pages for more information.

We also work closely with the Cyberinfrastructure Development (CyD) group within CCT. The CyD group can provide an even higher level of consultation by assisting with code development or help in porting existing codes to new frameworks.

Visualization
Visualization services are provided at the Visualization Lab, located at 101 Frey Computing Services Center. The lab is equipped with high end workstations and projection systems along with various powerful visualization software packages. It also has 3D capability with stereo displays, 3D glasses, 3D scanners and navigation tools. Basic user support (tier 1) is provided at the lab, while advanced user support (tier 2) is provided by CCT.
## Available HPC resources

<table>
<thead>
<tr>
<th>Name</th>
<th>Peak Performance (TFLOPS)</th>
<th>Location</th>
<th>Vendor</th>
<th>Architecture</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Queen Bee</td>
<td>50.7</td>
<td>ISB</td>
<td>Dell</td>
<td>Linux x86_64</td>
<td>In production</td>
</tr>
<tr>
<td>Eric</td>
<td>4.8</td>
<td>LSU</td>
<td>Dell</td>
<td>Linux x86_64</td>
<td>In production</td>
</tr>
<tr>
<td>Oliver</td>
<td>4.8</td>
<td>ULL</td>
<td>Dell</td>
<td>Linux x86_64</td>
<td>In production</td>
</tr>
<tr>
<td>Louie</td>
<td>4.8</td>
<td>Tulane</td>
<td>Dell</td>
<td>Linux x86_64</td>
<td>In production</td>
</tr>
<tr>
<td>Poseidon</td>
<td>4.8</td>
<td>UNO</td>
<td>Dell</td>
<td>Linux x86_64</td>
<td>In production</td>
</tr>
<tr>
<td>Painter</td>
<td>4.8</td>
<td>LaTech</td>
<td>Dell</td>
<td>Linux x86_64</td>
<td>In production</td>
</tr>
<tr>
<td>Satellite</td>
<td>4.8</td>
<td>Southern</td>
<td>Dell</td>
<td>Linux x86_64</td>
<td>Being deployed?</td>
</tr>
<tr>
<td>Tezpur</td>
<td>15.3</td>
<td>LSU</td>
<td>Dell</td>
<td>Linux x86_64</td>
<td>In production</td>
</tr>
<tr>
<td>Philip</td>
<td>3.5</td>
<td>LSU</td>
<td>Dell</td>
<td>Linux x86_64</td>
<td>In production</td>
</tr>
<tr>
<td>Pandora</td>
<td>6.8</td>
<td>LSU</td>
<td>IBM</td>
<td>Power7</td>
<td>In production</td>
</tr>
<tr>
<td>SuperMikell</td>
<td>146(CPU)+66(GPU)</td>
<td>LSU</td>
<td>Dell</td>
<td>Linux x86_64</td>
<td>In production</td>
</tr>
<tr>
<td>SuperMIC</td>
<td>~1000</td>
<td>LSU</td>
<td>Dell</td>
<td>Linux x86_64</td>
<td>Arriving</td>
</tr>
</tbody>
</table>
# LSU HPC Resources

## SuperMike II
- **Hostname**: mike.hpc.lsu.edu
- **Peak Performance/TFlops**: 146
- **Compute nodes**: 440
- **Processor/node**: 2 Octa-core
- **Processor Speed**: 2.6GHz
- **Processor Type**: Intel Xeon 64-bit
- **Nodes with Accelerators**: 50
- **Accelerator Type**: 2 nVidia M2090
- **OS**: RHEL v6
- **Vendor**: Dell
- **Memory per node**: 32/64/256 GB

## Pandora
- **Hostname**: pandora.hpc.lsu.edu
- **Peak Performance/TFlops**: 6.8
- **Compute nodes**: 8
- **Processor/node**: 32 (4 threads each)
- **Processor Speed**: 3.3GHz
- **Processor Type**: IBM POWER7
- **Nodes with Accelerators**: 0
- **Accelerator Type**:
- **OS**: AIX v7.1
- **Vendor**: IBM
- **Memory per node**: 128 GB

## Philip
- **Hostname**: philip.hpc.lsu.edu
- **Peak Performance/TFlops**: 3.469
- **Compute nodes**: 37
- **Processor/node**: 2 Quad-Core
- **Processor Speed**: 2.93GHz
- **Processor Type**: Intel Xeon 64-bit
- **Nodes with Accelerators**: 2
- **Accelerator Type**: 3 nVidia M2070
- **OS**: RHEL v5
- **Vendor**: Dell
- **Memory per node**: 24/48/96 GB

---

2/5/14

HPC User Environment Spring 2014
Why would you use SuperMike II?

- You need many nodes with more cores
  - 16 cores, 32G/node
- You need special nodes
  - Memory > 200G
  - GPUs on the node
- You need special storage
  - /project

Why would you use Pandoa?

- You need an AIX cluster/IBM processors
- You need many cores/memory on one node. For threaded non-mpi jobs
  - 128G/node
  - 32 thds@3.3 GHz/ nodes

Why would you use Philip?

- You need medium memory, fast single core for serial jobs
  - 24-96G, 8 cores @2.93GHz / node
- You need shared storage with SuperMike-II
  - /project not shared with SM-II. Earlier with Tezpur
LSU HPC: Watch out for SuperMIC

360 Compute Nodes
- Two 2.8GHz 10-Core Ivy Bridge-EP E5-2680 Xeon 64-bit Processors
- Two Intel Xeon Phi 7120P Coprocessors
- 64GB DDR3 1866MHz Ram
- 500GB HD
- 56 Gigabit/sec Infiniband network interface

20 Hybrid Compute Nodes
- Two 2.8GHz 10-Core Ivy Bridge-EP E5-2680 Xeon 64-bit Processors
- One Intel Xeon Phi 7120P Coprocessors
- One NVIDIA Tesla K20X 6GB GPU with GPUDirect Support
- 64GB DDR3 1866MHz Ram
- 500GB HD
- 56 Gigabit/sec Infiniband network interface

Cluster Storage
- 840TB Lustre High-Performance disk
- 5TB NFS-mounted /home disk storage
**ssh: Accessing the cluster**

- **Host name**
  - LONI: `<cluster name>.loni.org` *e.g.: mike.hpc.lsu.edu*
  - LSU HPC: `<cluster name>.hpc.lsu.edu` *e.g.: qb.loni.org*
  - On Unix and Mac use `ssh` on a terminal to connect

```bash
$ ssh bthakur@mike.hpc.lsu.edu
bthakur@mike.hpc.lsu.edu's password:
Last login: Tue Jul 9 21:41:24 2013 from i###

Send questions and comments to the email ticket system at sys-help@loni.org.

SuperMike-II at LSU (Open for general use)

1-Dec-2012

SuperMike-II is a 146 TFlops Peak Performance, 440 node, 16 processor Red Hat Enterprise Linux 6 cluster from Dell with 2.6 GHz Intel Xeon 64-bit processors and 32 GB RAM per node. GPUs and additional memory are available on some nodes. This cluster is for authorized users of the LSU community. Access is restricted to those who meet the criteria as stated on our website.
Accessing the Clusters

- Host name
  - LONI: ssh <cluster name>.loni.org  e.g.: mike.hpc.lsu.edu
  - LSU HPC: ssh <cluster name>.hpc.lsu.edu  e.g.: qb.loni.org
  - On Windows use putty
Connection with X11 Forwarding

- Some software packages have GUI, which requires X11 forwarding to be established with the ssh connection
- Unix/Linux users
  - Use the “-X” option of ssh
- Mac users
  - Use the “X11” application
- Windows users
  - Install X server (e.g. Xming)
  - Enable X11 forwarding in the client
## File Systems

<table>
<thead>
<tr>
<th></th>
<th>Distributed</th>
<th>Throughput</th>
<th>life</th>
<th>Best used for</th>
</tr>
</thead>
<tbody>
<tr>
<td>Home</td>
<td>Yes</td>
<td>Low</td>
<td>Unlimited</td>
<td>Development/compilation</td>
</tr>
<tr>
<td>Work/Scratch</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>Higher?</td>
<td>Job duration</td>
<td>Temporary files</td>
</tr>
</tbody>
</table>

- **Tips**
  - Never let your job write output to your home directory
  - Do not write temporary files to `/tmp`. Write to local scratch or work space
  - The work space is not for long-term storage. Files purged periodically
  - Use "rmpurge" to delete large amount of files
Disk Quota

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Home</th>
<th>Work</th>
<th>Local scratch</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Access point</td>
<td>Quota</td>
<td>Access Point</td>
</tr>
<tr>
<td>LONI Linux</td>
<td>/home/$USER</td>
<td>5 GB</td>
<td>/work/$USER</td>
</tr>
<tr>
<td>HPC Linux</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HPC AIX</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- No quota is enforced on the work space on Queen Bee, Tezpur, Philip and SuperMikell
- On Linux clusters, the work directory is created within an hour after the first login
- Check current disk usage
  - **Linux**: `showquota`
Storage Allocation on /project

• One can apply for extra disk space on the /project volume if
  – your research requires some files to remain on the cluster for a fairly long period of time; \textbf{and}
  – their size exceeds the quota of the /home
• The unit is 100 GB
• Available on SuperMikelll and Queen Bee
• Storage allocations are good for 6 months, but can be extended based on the merit of the request
• Examples of valid requests
  – I am doing a 6-month data mining project on a large data set
  – The package I am running requires 10 GB of disk space to install
• Examples of invalid requests
  – I do not have time to transfer the data from my scratch space to my local storage and I need a temporary staging area
File Transfer

From/to a Unix/Linux/Mac machine

Use scp or rsync

```
scp <options> <source> <destination>
rsync <options> <source> <destination>
```

```
$ scp
usage: scp [-1246BCEpqrv] [-c cipher] [-F ssh_config] [-i identity_file]
    [-l limit] [-o ssh_option] [-P port] [-S program]
    [[user@]host1:]file1 ... [[user@]host2:]file2
```
File Transfer

From a Windows machine

• Use a client that supports the scp protocol (e.g. WinSCP, Filezilla)
Software

- Learn to use softenv
- Know your compilers
- Find your applications or port your stuff and setup your simulation
Using softenv

Environment variables
- PATH: where to look for executables
- LD_LIBRARY_PATH: where to look for shared libraries
- LDINCLUDE_PATH: where to look for header and include files

Other environment variables sometimes needed by various software
- LIBRARY_PATH, C_LIBRARY_PATH
- LDFLAGS, LDLIBS

SOFTENV is a software that helps users set up environment variables properly to use other software package. Much more convenient than setting variables in .bashrc

Modules is another software that helps users set up their environment. Most supercomputing sites have moved onto modules. We are also planning to move to modules with newer machines
Listing All Packages

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

Softenv on SuperMikell shown here

Softenv key

---

SoftEnv version 1.6.2
The SoftEnv system is used to set up environment variables. For details, see 'man softenv-intro'.

This is a list of keys and macros that the SoftEnv system understands. In this list, the following symbols indicate:
* This keyword is part of the default environment, which you get by putting "@default" in your .soft
U This keyword is considered generally "useful".
P This keyword is for "power users", people who want to build their own path from scratch. Not recommended unless you know what you are doing.

These are the macros available:
* @default

These are the keywords explicitly available:
+ImageMagick-6.7.9-gcc-4.4.6 @Types: Application/Visualization @name: ImageMagick @version: 6.7.9 @build: ImageMagick-6.7.9-gcc-4.4.6 @internal: @external: http://www.imagemagick.org @about: A software suite to create, edit, and compose bitmap images.
+Intel-12.1.4 @Types: Programming/Compiler @name: Intel @version: 12.1.4 @build: Binary @installation @internal: @external: http://software.intel.com/en-us/articles/intel-compilers/ @about: The C/C++ and Fortran compiler suite from Intel.
+Intel-13.0.0 @Types: Programming/Compiler @name: Intel @version: 13.0.0 @build: Binary @installation @internal: @external: http://software.intel.com/en-us/articles/intel-compilers/ @about: The C/C++ and Fortran compiler suite from Intel.
Searching A Specific Package

Use “–k” option with softenv command to search a particular key
Searching A Specific Package

Use “–k” option with softenv command to search a key.

You can also grep
$ softenv |grep "openmpi"

<table>
<thead>
<tr>
<th>Search Regexp: fftw</th>
</tr>
</thead>
<tbody>
<tr>
<td>These are the macros available:</td>
</tr>
<tr>
<td>+fftw-3.3.2-Intel-13.0.0 @types: Library/Math @name: fftw @version: 3.3.2 @build: Intel-13.0.0 @Internal: @external: <a href="http://www.fftw.org">www.fftw.org</a> @about: A fast, free C FFT library; includes real-complex, multidimensional, and parallel transforms.</td>
</tr>
<tr>
<td>+fftw-3.3.3-Intel-13.0.0 @types: Library/Math @name: fftw @version: 3.3.3 @build: Intel-13.0.0 @Internal: @external: <a href="http://www.fftw.org">www.fftw.org</a> @about: A fast, free C FFT library; includes real-complex, multidimensional, and parallel transforms.</td>
</tr>
<tr>
<td>+fftw-3.3.3-Intel-13.0.0-openmpi-1.6.2 @types: Library/Math @name: fftw @version: 3.3.3 @build: Intel-13.0.0-openmpi-1.6.2 @Internal: @external: <a href="http://www.fftw.org">www.fftw.org</a> @about: A fast, free C FFT library; includes real-complex, multidimensional, and parallel transforms.</td>
</tr>
</tbody>
</table>

-bash-4.1 @ mike1$ softenv |grep "openmpi"

openmpi-1.6.2 @internal:
openmpi-1.6.2 @internal:
openmpi-1.6.2 @internal:
openmpi-1.6.2-CUDA-4.2.9 @internal:
openmpi-1.6.2-CUDA-4.2.9 @internal:
openmpi-1.6.2-CUDA-4.2.9 @internal:
openmpi-1.6.2-openmpi-1.6.2 @internal:
openmpi-1.6.2-openmpi-1.6.2 @internal:
openmpi-1.6.2-openmpi-1.6.2 @internal:
openmpi-1.6.2-openmpi-1.6.2 @internal:
openmpi-1.6.2-openmpi-1.6.2 @internal:
openmpi-1.6.2-openmpi-1.6.2 @internal:
Setting up Environment via Softenv:

One time change

Set up the environment to use a package **in the current session only**

- Add a package: `soft add <key>`
- Remove a package: `soft delete <key>`

```bash
$ which gcc
/usr/bin/gcc

$ softenv | grep "+gcc"
+gcc-4.7.2

$ soft add +gcc-4.7.2

$ which gcc
/usr/local/compilers/GNU/gcc-4.7.2/bin/gcc

$ soft delete +gcc-4.7.2

$ which gcc
/usr/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

```
$ which python
/usr/bin/python

$ cat ~/.soft
#
+Python-2.7.3-gcc-4.4.6
+fftw-3.3.3-Intel-13.0.0
+cuda-4.2.9
@default

$ resoft
$ which python
/usr/local/packages/Python/2.7.3/gcc-4.4.6/bin/python
```
“soft-dbq” : Querying a Softenv key

-bash-4.1 @ mike1$ soft-dbq +gcc-4.7.2
This is all the information associated with
the key or macro +gcc-4.7.2.
-------------------------------------------
Name: +gcc-4.7.2
Description:
@types: Programming/Compiler
@name: gcc-4.7.2
@version: 4.7.2
@internal:
@external: http://gcc.gnu.org
@about: Free compilers from GNU
Flags: none Groups: noneExists on: Linux
-------------------------------------------
On the Linux architecture,
the following will be done to the environment:
The following environment changes will be made:
  GCC_HOME = /usr/local/compilers/GNU/gcc-4.7.2
  LD_INCLUDE_PATH = ${LD_INCLUDE_PATH}:/usr/local/compilers/GNU/gcc-4.7.2/include
  LD_LIBRARY_PATH = ${LD_LIBRARY_PATH}:/usr/local/compilers/GNU/gcc-4.7.2/lib64
  MANPATH = ${MANPATH}:/usr/local/compilers/GNU/gcc-4.7.2/man
  PATH = ${PATH}:/usr/local/compilers/GNU/gcc-4.7.2/bin
“soft-dbq” : Querying a Softenv key

Do not remove the @default key

$ soft-dbq @default
This is all the information associated with the key or macro @default.
Name: @default
Description: No description set.
Flags: none
Groups: none
Exists on: Linux aix-5 aix-53 linux linux-sles8-ia64 solaris-9
-------------------------------------------
@default contains the following keywords and macros:
+Intel-13.0.0 +openmpi-1.6.2-Intel-13.0.0 +default
-------------------------------------------

$ soft-dbq +default
This is all the information associated with the key or macro +default.
Name: +default
Description: No description set.
Flags: none
Groups: none
Exists on: Linux aix-5 aix-53 linux linux-sles8-ia64 solaris-9
-------------------------------------------
On the Linux architecture, the following will be done to the environment:
The following environment changes will be made:
ARCH = `${WHATAMI}`
MANPATH = `${MANPATH}:/usr/X11R6/man:/usr/share/man:
/usr/share/locale/en/man:/usr/bin/man:/usr/local/share/man:
/usr/local/man:/usr/local/man:/usr/local/packages/softenv/man
PATH = `${PATH}:/bin:/usr/bin:/sbin:/usr/sbin:/usr/local/bin:
/usr/local/sbin:/usr/X11R6/bin:/usr/local/packages/softenv/bin
PLATFORM = `${WHATAMI}`
WHATAMI = `/usr/local/packages/softenv/bin/whatami`
Using softenv: Quiz

$ cat ~/.soft
#
+openmpi-1.6.2-gcc-4.7.2
@default

Which mpif90/mpirun will the system use if you just call mpif90/mpirun?
Which compiler will be used?
Using softenv: Quiz

Which mpif90/mpirun will the system use if you just call mpirun?

$ cat ~/.soft
#
+mvapich2-1.8.1-Intel-13.0.0
@default
+openmpi-1.6.2-gcc-4.7.2

HPC User Environment Spring 2014
Using softenv: Quiz

Which mpif90/mpirun will the system use if you just call mpirun?

```
$ cat ~/.soft
# PATH += /usr/local/packages/mpich/3.0.2/Intel-13.0.0/bin
+mvapich2-1.8.1-Intel-13.0.0
@default
+openmpi-1.6.2-gcc-4.7.2
```
Which version of intel fortran compiler will be displayed by the commands “mpif90 –version” ?
Exercise : Use Softenv

- Find the key for Python 2.7.3
- Check what variables are set through the key
- Set up your environment to use Python 2.7.3
- Check if the variables are correctly set by “which python”
- Check if you have access to ipython, scipy, numpy, matplotlib
## Compilers

<table>
<thead>
<tr>
<th>Language</th>
<th>Linux cluster</th>
<th>AIX clusters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel</td>
<td>PGI</td>
<td>GNU</td>
</tr>
<tr>
<td></td>
<td></td>
<td>XL</td>
</tr>
<tr>
<td>Fortran</td>
<td>ifort</td>
<td>pgf77, pgf90</td>
</tr>
<tr>
<td></td>
<td></td>
<td>xlf, xlf90</td>
</tr>
<tr>
<td>C</td>
<td>icc</td>
<td>pgcc</td>
</tr>
<tr>
<td></td>
<td></td>
<td>xlc</td>
</tr>
<tr>
<td>C++</td>
<td>icpc</td>
<td>pgCC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>xlC</td>
</tr>
</tbody>
</table>

### Serial compilers

<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>mpxlf, mpxlf90</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>mpicc</td>
<td>mpcc</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C++</td>
<td>mpiCC</td>
<td>mpCC</td>
</tr>
</tbody>
</table>

### Parallel compilers
To compile the program, use any

$ ifort test_hello2.f90
$ gfortran test_hello2.f90

To verify which compiler was used

$ nm a.out | grep -i intel
... __intel_cpu_indicator

$ nm -s a.out | grep -i gfortran
... _gfortran_cpu_time_4@...
To compile the program, use any

$ gcc test_hello2.c –lrt
$ icc test_hello2.c -lrtc

Cpu vs Elapsed time

$ ./a.out
Value 501446
Time Cpu 0.320000
Time Elp 10.326020

$ ./a.out
Value 501446
Time Cpu 0.190000
Time Elp 10.198743
To compile the program, use any

$ ifort -openmp test_hello3.f90
$ gfortran -fopenmp test_hello3.f90

Verify execution with intel

$ export OMP_NUM_THREADS=16
$ ./a.out
Value 3.9361696E+08
Time Elapsed 0.1250000
Time Cpu 1.935706
Utilization 0.9678530
Exercise: threaded C code

Parallelize this code with OpenMP

1. Put in openmp directives
2. Compile with additional openmp flags

This might be tough if you are not used to programming

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main()
{
    clock_t t0, t1;
    struct timespec t2, t3;

    t0 = clock();
    clock_gettime(CLOCK_REALTIME, &t2);

    int i, j; float r = 0.0f;
    for (i = 1; i < 100000; i++)
    {
        for (j = 1; j < 10000; j++)
        {
            if ((i % 11 == 0) && (j % 13 == 0))
            {
                r = r + (i * 13.0f) / (j * 11.0f);
            }
        }
    }

    t1 = clock();
    clock_gettime(CLOCK_REALTIME, &t3);

    float etime = (float)(t3.tv_sec + t3.tv_nsec * 1e-9) - \
                   (t2.tv_sec + t2.tv_nsec * 1e-9);
    float ctime = (float)(t1 - t0) / CLOCKS_PER_SEC;

    printf("Value \%g\n", r);
    printf("Time Cpu \%f\n", ctime);
    printf("Time Elp \%f\n", etime);

    return 0;
}
```
## MPI libraries

<table>
<thead>
<tr>
<th>Cluster Resource</th>
<th>Name</th>
<th>Mvapich</th>
<th>Mvapich2</th>
<th>Openmpi</th>
<th>mpich</th>
<th>Default serial compiler</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONI</td>
<td>Queen Bee</td>
<td>.98, 1.1</td>
<td>1.4, 1.6, 1.8.1</td>
<td>1.3.4</td>
<td>X</td>
<td>Intel 11.1</td>
</tr>
<tr>
<td></td>
<td>Other LONI</td>
<td>.98, 1.1</td>
<td>1.4, 1.6</td>
<td>1.3.4</td>
<td>X</td>
<td>Intel 11.1</td>
</tr>
<tr>
<td>LSU</td>
<td>Tezpur</td>
<td>.98, 1.1</td>
<td>1.4, 1.6</td>
<td>1.3.4</td>
<td>X</td>
<td>Intel 11.1</td>
</tr>
<tr>
<td></td>
<td>Philip</td>
<td>X</td>
<td>X</td>
<td>1.4.3, 1.6.1</td>
<td>1.2.7, 1.3.2, 1.4.1</td>
<td>Intel 11.1</td>
</tr>
<tr>
<td></td>
<td>SuperMikellII</td>
<td>X</td>
<td>1.6, 1.9</td>
<td>1.6.x, 1.9ax</td>
<td>3.0.x</td>
<td>Intel 13.0.0</td>
</tr>
<tr>
<td></td>
<td>Pandora</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>AIX</td>
</tr>
</tbody>
</table>
## MPI Compilers

<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>mpxlf, mpxlf90</td>
</tr>
<tr>
<td>C</td>
<td>mpicc</td>
<td>mpcc</td>
</tr>
<tr>
<td>C++</td>
<td>mpiCC</td>
<td>mpCC</td>
</tr>
</tbody>
</table>

mpif90 hello.f90

mpicc hello.c

mpicxx hello.cpp
Compiling a MPI C program

Compiling Hello world:

```c
#include <stdio.h>
#include "mpi.h"

int main( argc, argv )
int argc;
char **argv;
{
    int rank, size;
    MPI_Init( &argc, &argv );
    MPI_Comm_size( MPI_COMM_WORLD, &size );
    MPI_Comm_rank( MPI_COMM_WORLD, &rank );
    printf( "Hello from process %d of %d\n", rank, size );
    MPI_Finalize();
    return 0;
}
```
Compiling a MPI Fortran program

Compiling Hello world:

mpif90 hello_more.f90
Always verify what compiler/library is being used

$ mpicc -show
icc -l/usr/local/packages/openmpi/1.6.2/Intel-13.0.0/include
   -L/usr/local/packages/openmpi/1.6.2/Intel-13.0.0/lib
   -lmpi -ldl -lm -Wl,--export-dynamic -lrt -lnsl
   -libverbs -libumad -lpthread -lutil

$ mpif90 -show
ifort -l/usr/local/packages/openmpi/1.6.2/Intel-13.0.0/include
   -L/usr/local/packages/openmpi/1.6.2/Intel-13.0.0/lib
   -lmpi_f90 -lmpi_f77 -lmpi
   -ldl -lm -Wl,--export-dynamic -lrt -lnsl -libverbs -libumad -lpthread -lutil
Compiling a MPI program

Always verify what library is being used: Before and after!

$ ldd a.out

...  
libmpi_f90.so.1 => /usr/local/packages/openmpi/1.6.2/Intel-13.0.0/lib/
libmpi_f90.so.1 (0x00002ba516b000)
libmpi.so.1 => /usr/local/packages/openmpi/1.6.2/Intel-13.0.0/lib/
libmpi.so.1 (0x00002ba55a6000)
libibverbs.so.1 => /usr/lib64/libibverbs.so.1 (0x0000003ec5c00000)
...  
libpthread.so.0 => /lib64/libpthread.so.0 (0x0000003e53e00000)
...  
libifport.so.5 => /usr/local/compiler/Intel/compiler_xe_2013.0.079/  
compiler/lib/intel64/libifport.so.5 (0x00002ba5fbbdb000)
Analysing a parallel( mpi ) program

Running a mpi program:
A process perspective
Analyzing a Hybrid parallel program

Compiling Hybrid Hello world:

mpif90 –openmp hello_hybrid.f90

```fortran
! Init MPI
    call MPI_Init(mpierr)

! Get Rank Size
    call MPI_COMM_Rank(MPI_COMM_WORLD, nrank, mpierr)
    call MPI_COMM_Size(MPI_COMM_WORLD, nproc, mpierr)

! Print rank
    call MPI_GET_PROCESSOR_NAME(pname, nlen, mpierr)

! Get Date hostname etc
    if (nrank==0) then
      call system('hostname && date && echo rank-pid $$
      end if
    call MPI_Barrier(MPI_COMM_WORLD, mpierr)

! OpenMP
  !$OMP PARALLEL PRIVATE(itd,gtd)
  itd= omp_get_thread_num()
  gtd= omp_get_num_threads()
  grank= nrank*gtd + itd
  write(*,'(4(a6,i6),a2,a8)')"Gid ", grank, " Im ", nrank,
  " of ", nproc,
  " thd", itd,
  " on ", pname

  !$OMP FLUSH
  !$OMP BARRIER
  if (nrank==0 .and. itd==0) then
    call system('pstree -ap -u bthakur ')
```
Analyzing a Hybrid parallel program

Running a hybrid (mpi + openmp) process
Application Software

Broadly we can classify them as

- Programming Language: Fortran, C, C++, CUDA
- Serial Compilers
- Parallel compiler/Library
- Serial/threaded application
- Dynamic/Interpreted Language
- Matlab
- Python/Perl/Ruby/BASH
- Applications
- Domain science Application
- Profiling/Debugging
- I/O
- Throughput, e.g. Bioinformatics
- Grid applications
- MPI
- Applications

HPC User Environment Spring 2014
Application Software

- **List of software**
  
  
  /usr/local/packages and /usr/local/compilers

  Run softenv

- **Installed Software**
  
  Numerical, I/O libraries: Lapack, FFTW, HDF5, NetCDF, PETSc
  
  Molecular Dynamics: Amber, Gromacs, NAMD, LAMMPS...
  
  Programming Tools: Totalview, DDT, TAU
  
  Licensed: Matlab, Fluent

- **User requested packages**
  
  Usually installed in user space, unless request by a group of users, in which case it will be installed under /usr/local/packages
Exercises

1. Serial:
   Compare the speed of serial C code with Intel, GCC and PGI compiler.
   Can you tune the compile options to produce best timing?

2. OpenMP:
   Modify the serial C code to be OMP threaded.
   Find compile time option for creating threaded cide with PGI compiler (pgcc)
   Compare performance vs Intel and GCC compilers

3. MPI:
Job management

- Job management basics
  - Find appropriate queue
  - Understand the queuing system and your requirements and proceed to submit jobs
  - Monitor jobs
Job Queues

- Nodes are organized into queues. Nodes can be shared.
  Each job queue differs in
  - Number of available nodes
  - Max run time
  - Max running jobs per user
  - Nodes may have special characteristics: GPU’s, Large memory etc

- Jobs need to specify resource requirements
  - Nodes, time, queue

- It’s called a queue for a reason, but jobs don’t run on a ‘First come first served’ policy.
## Queue Characteristics – LONI clusters

<table>
<thead>
<tr>
<th>Machine</th>
<th>Queue</th>
<th>Max Runtime</th>
<th># of nodes</th>
<th>Max running jobs per user</th>
<th>Max nodes per job</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Queen Bee</td>
<td>workq</td>
<td>3 days</td>
<td>530</td>
<td>8</td>
<td>128</td>
<td>Unpreemptable</td>
</tr>
<tr>
<td></td>
<td>checkpoint</td>
<td></td>
<td>668</td>
<td></td>
<td>256</td>
<td>Preemptable</td>
</tr>
<tr>
<td>Others</td>
<td>workq</td>
<td>3 days</td>
<td>128</td>
<td>8</td>
<td>40</td>
<td>Unpreemptable</td>
</tr>
<tr>
<td></td>
<td>checkpoint</td>
<td></td>
<td>96</td>
<td></td>
<td>64</td>
<td>Preemptable</td>
</tr>
<tr>
<td></td>
<td>single</td>
<td>14 days</td>
<td>16</td>
<td>64</td>
<td>1</td>
<td>Single processor</td>
</tr>
</tbody>
</table>
# Queue Characteristics – LSU Linux clusters

<table>
<thead>
<tr>
<th>Machine</th>
<th>Queue</th>
<th>Max Runtime</th>
<th># of nodes</th>
<th>Max running jobs per user</th>
<th>Max nodes per job</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>SuperMikell</td>
<td>workq</td>
<td>3 days</td>
<td>128</td>
<td>48</td>
<td>128</td>
<td>Unpreemptable</td>
</tr>
<tr>
<td></td>
<td>checkpoint</td>
<td>2 days</td>
<td>96</td>
<td></td>
<td>200</td>
<td>Preemptable</td>
</tr>
<tr>
<td></td>
<td>bigmem</td>
<td>1 day</td>
<td>8</td>
<td></td>
<td>2</td>
<td>Big memory</td>
</tr>
<tr>
<td></td>
<td>gpu</td>
<td>2 days</td>
<td>50</td>
<td></td>
<td>32</td>
<td>Job using GPU</td>
</tr>
<tr>
<td>Tezpur</td>
<td>workq</td>
<td>3 days</td>
<td>180</td>
<td>8</td>
<td>90</td>
<td>Unpreemptable</td>
</tr>
<tr>
<td></td>
<td>checkpoint</td>
<td>2 days</td>
<td>344</td>
<td></td>
<td>180</td>
<td>Preemptable</td>
</tr>
<tr>
<td></td>
<td>single</td>
<td>14 days</td>
<td>16</td>
<td>64</td>
<td>1</td>
<td>Single processor</td>
</tr>
<tr>
<td>Philip</td>
<td>workq</td>
<td>3 days</td>
<td>28</td>
<td>12</td>
<td>5</td>
<td>Unpreemptable</td>
</tr>
<tr>
<td></td>
<td>checkpoint</td>
<td>2 days</td>
<td>28</td>
<td></td>
<td>5</td>
<td>Preemptable</td>
</tr>
<tr>
<td></td>
<td>gpu</td>
<td>1 day</td>
<td>2</td>
<td></td>
<td>1</td>
<td>Job using GPU</td>
</tr>
<tr>
<td></td>
<td>bigmem</td>
<td>14 days</td>
<td>5</td>
<td></td>
<td>1</td>
<td>Big memory</td>
</tr>
<tr>
<td></td>
<td>single</td>
<td>14 days</td>
<td>24</td>
<td></td>
<td>1</td>
<td>Single processor</td>
</tr>
</tbody>
</table>
Queue Characteristics – LSU AIX Clusters

<table>
<thead>
<tr>
<th>Machine</th>
<th>Queue</th>
<th>Max Runtime</th>
<th># of cores</th>
<th>Max running jobs per user</th>
<th>Max cores per job</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pandora</td>
<td>Interactive</td>
<td>30 minutes</td>
<td>8</td>
<td>6</td>
<td>8</td>
<td>Unpreemptable</td>
</tr>
<tr>
<td></td>
<td>Workq</td>
<td>3 days</td>
<td>224</td>
<td>6</td>
<td>128</td>
<td>Preemptable</td>
</tr>
<tr>
<td></td>
<td>Single</td>
<td>7 days</td>
<td>64</td>
<td></td>
<td>32</td>
<td>Single processor</td>
</tr>
</tbody>
</table>
Queue Characteristics

“qstat –q” will give you more info on the queues

For a more detailed description use mdiag
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

-bash-4.1 @ mike1$ qfree
PBS total nodes: 453, free: 106, busy: 315 *12, down: 32, use: 69%
PBS workq nodes: 250, free: 3, busy: 89, queued: 35
PBS checkpt nodes: 290, free: 3, busy: 189, queued: 78
PBS lasigma nodes: 30, free: 0, busy: 29, queued: 1
PBS mwfa nodes: 8, free: 0, busy: 7, queued: 1
PBS single nodes: 10, free: 0 *12, busy: 1, queued: 0
(Highest priority job 33426 on queue workq will start in 2:59:50)

-bash-3.00 @ qb3$ qfree
PBS total nodes: 668, free: 29, busy: 630, down: 9, use: 94%
PBS workq nodes: 529, free: 23, busy: 309, queued: 253
PBS checkpt nodes: 656, free: 26, busy: 321, queued: 76
(Highest priority job 699177 on queue checkpt will start in 2:58:51)
Resource managers give access to compute resource

- Takes in a request on login node
- Finds appropriate resource and assigns you a priority number
- Positions your job in a queue based on the priority assigned.
- Starts running jobs until it cannot run more jobs with what is available.

Note
- Newer jobs coming in can have a higher priority as it follows a complex calculation for priority number
Resource manager philosophy

- Prioritize workload into a queue for jobs
- Backfill idle nodes to maximize utilization
Job priorities

Job priorities have contributions from the following

- Resource requirements.
- Time spent in queue
- User Credentials
- Fair-share

“qstat –a” to see what’s running/queued
Don’t run it too often as it an intensive query
“qstat –u $USER” to see your jobs
Backfilling

Backfilling aims to utilize idle nodes by running jobs out of order. Enabling backfill allows the scheduler to start other, lower-priority jobs so long as they do not delay the highest priority job.

If the **FIRSTFIT** algorithm is applied, the following steps are taken:

- The list of feasible backfill jobs is filtered, selecting only those that will actually fit in the current backfill window.
- The first job is started.
- While backfill jobs and idle resources remain, repeat step 1.
Backfilling aims to utilize idle nodes by running jobs out of order. Enabling backfill allows the scheduler to start other, lower-priority jobs so long as they do not delay the highest priority job.

Although the highest priority job is protected, there is nothing to prevent the third priority job from starting early and possibly delaying the start of the second priority job.

showbf will show you the current backfill windows
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
    - Do not run on the head node !!!
    - Try not to run interactive jobs with large core count, which is a waste of resources)

- **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 -l -V \
  -l walltime=<hh:mm:ss>,nodes=<num nodes>:ppn=<num cores> \
  -A <Allocation> \
  -q <queue name>
  ```

- Batch job
  ```
  qsub job_script
  ```

- Add -X to enable X11 forwarding
#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 <allocation_if_needed>
#PBS -m e
#PBS -M <email address>

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

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

<shell commands>
<path_to_executable> <options>
<shell commands>
Check details on your job using `qstat`

\[ $ \textit{qstat} -f \textit{jobid} \] : For details on your job

\[ $ \textit{qstat} -n -u \$\textit{USER} \] : For quick look at nodes assigned to you

\[ $ \textit{qdel} \textit{jobid} \] : To delete job

Check approximate start time using `showstart`

\[ $ \textit{showstart} \textit{jobid} \]

Check details of your job using `checkjob`

\[ $ \textit{checkjob} \textit{jobid} \]

Check health of your job using `qshow`

\[ $ \textit{qshow} \textit{--j} \textit{jobid} \]

Pay close attention to the load and the memory consumed by your job.
## Queue Querying – AIX Clusters

- **Command:** `llclass`

```bash
eyan1@l2f1n03$ 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    56  priority queue reserved for on-demand jobs (5 days), up to 48 processors
preempt    unlimited    unlimited    40    56  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
```

---
LoadLeveler Job Script - Parallel

```bash
#!/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
#@ total_tasks = 16
#@ requirements = (Arch == “POWER7”)
#@ environment = COPY_ALL
#@ queue
  <shell commands>
poe <path_to_executable> <options>
  <shell commands>
```

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>#!/bin/sh</code></td>
<td>Job type</td>
</tr>
<tr>
<td><code>#@ job_type = parallel</code></td>
<td>Standard output</td>
</tr>
<tr>
<td><code>#@ output = /work/default/username/$(jobid).out</code></td>
<td>Standard output</td>
</tr>
<tr>
<td><code>#@ error = /work/default/username/$(jobid).err</code></td>
<td>Standard error</td>
</tr>
<tr>
<td><code>#@ notify_user = youremail@domain</code></td>
<td>Notification</td>
</tr>
<tr>
<td><code>#@ notification = error</code></td>
<td>Notify on error</td>
</tr>
<tr>
<td><code>#@ class = checkpt</code></td>
<td>Queue</td>
</tr>
<tr>
<td><code>#@ wall_clock_limit = 24:00:00</code></td>
<td>Wall clock time</td>
</tr>
<tr>
<td><code>#@ node_usage = shared</code></td>
<td>node usage</td>
</tr>
<tr>
<td><code>#@ node = 2</code></td>
<td># of nodes</td>
</tr>
<tr>
<td><code>#@ total_tasks = 16</code></td>
<td># of processors</td>
</tr>
<tr>
<td><code>#@ requirements = (Arch == “POWER7”)</code></td>
<td>Job requirement</td>
</tr>
<tr>
<td><code>#@ environment = COPY_ALL</code></td>
<td>Environment</td>
</tr>
</tbody>
</table>
Loadleveler Job Script - Serial

```bash
#!/bin/sh
@@ job_type = serial
@@ output = /work/default/username/$(jobid).out
@@ error = /work/default/username/$(jobid).err
@@ notify_user = youremail@domain
@@ notification = error
@@ class = single
@@ wall_clock_limit = 24:00:00
@@ requirements = (Arch == “POWER5”)
@@ environment = COPY_ALL
@@ queue

<shell commands>
poe <path_to_executable> <options>
<shell commands>
```
Submitting Jobs – AIX clusters

• Submit jobs using llsubmit
  – llsubmit jobscript : submit job
  – llcancel jobid : delete job

• Check job status using llq and cluster status using llstatus
Job Monitoring – AIX Clusters

- **Command:** `showllstatus.py`
  - Show job status and nodes running on
- **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>`

```
$ 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</td>
<td>R</td>
<td>50</td>
<td>single</td>
<td>l2f1n14</td>
</tr>
<tr>
<td>l2f1n03.3730.0</td>
<td>jheiko</td>
<td>1/28 13:30</td>
<td>R</td>
<td>50</td>
<td>workq</td>
<td>l2f1n10</td>
</tr>
<tr>
<td>l2f1n03.3726.0</td>
<td>collin</td>
<td>1/26 08:21</td>
<td>R</td>
<td>50</td>
<td>single</td>
<td>l2f1n14</td>
</tr>
<tr>
<td>l2f1n03.3698.0</td>
<td>collin</td>
<td>1/22 17:00</td>
<td>R</td>
<td>50</td>
<td>single</td>
<td>l2f1n14</td>
</tr>
<tr>
<td>l2f1n03.3727.0</td>
<td>collin</td>
<td>1/26 08:21</td>
<td>R</td>
<td>50</td>
<td>single</td>
<td>l2f1n14</td>
</tr>
</tbody>
</table>
```

5 job step(s) in queue, 0 waiting, 0 pending, 5 running, 0 held, 0 preempted
Exercise

Submit a small job to run “sleep 180” and “print PBS variables”

– Create a script to submit a 5 min job and print from within the job script PBS variables $PBS_NODEFILE, $PBS_WORKDIR. Also run “sleep 180” to give you a few minutes to verify status.

– Once the job is running, find out the Mother Superior node and other slave nodes assigned to your job using qstat.

– Log into MS node and verify that your job is running and find your temporary output file

– Modify your script to print hello from each of your assigned nodes

Run it within an interactive job session

– Verify using hostname that you are not on the head-node

– Check available PBS variables and print them

Run a shell script using mpirun to print process id of shell
Exercise

Run hello_hybrid.f90 as a batch job
  – On SM-II run on 2 nodes with 2 mpi-processes per node and 8 threads per mpi process.
  – On QB run 4 threads per mpi process
Future Trainings

• Weekly trainings during regular semester
  – Wednesdays “10am-12pm + afternoon” sessions, Frey 307

• Programming/Parallel Programming workshops
  – Usually in summer

Keep an eye on our webpage: www.hpc.lsu.edu