

# Introduction to LONI Computing Resources

Le Yan

Scientific computing consultant  
User services group  
LONI HPC





# Outline

- Hardware overview
- User environment
- Job management





# Outline

- Hardware overview
- User environment
- Job management



# LONI HPC clusters

- Two major platforms
  - Linux clusters
    - Vendor: Dell
    - Operating System: Linux (Red Hat Enterprise Linux)
    - Processor: Intel
  - AIX clusters
    - Vendor: IBM
    - Operating System: AIX
    - Processor: IBM



# Current deployment status - Dell Linux clusters

	Name	Peak TeraFLOPS	Location	Status	Login
LONI	Queen Bee	50.7	ISB	Available	LONI
	Eric	4.7	LSU	Available	LONI
	Oliver	4.7	ULL	Available	LONI
	Louie	4.7	Tulane	Available	LONI
	Poseidon	4.7	UNO	Available	LONI
	Painter	4.7	LaTech	Available	LONI

Manage your account:  
LONI <https://allocations.loni.org>



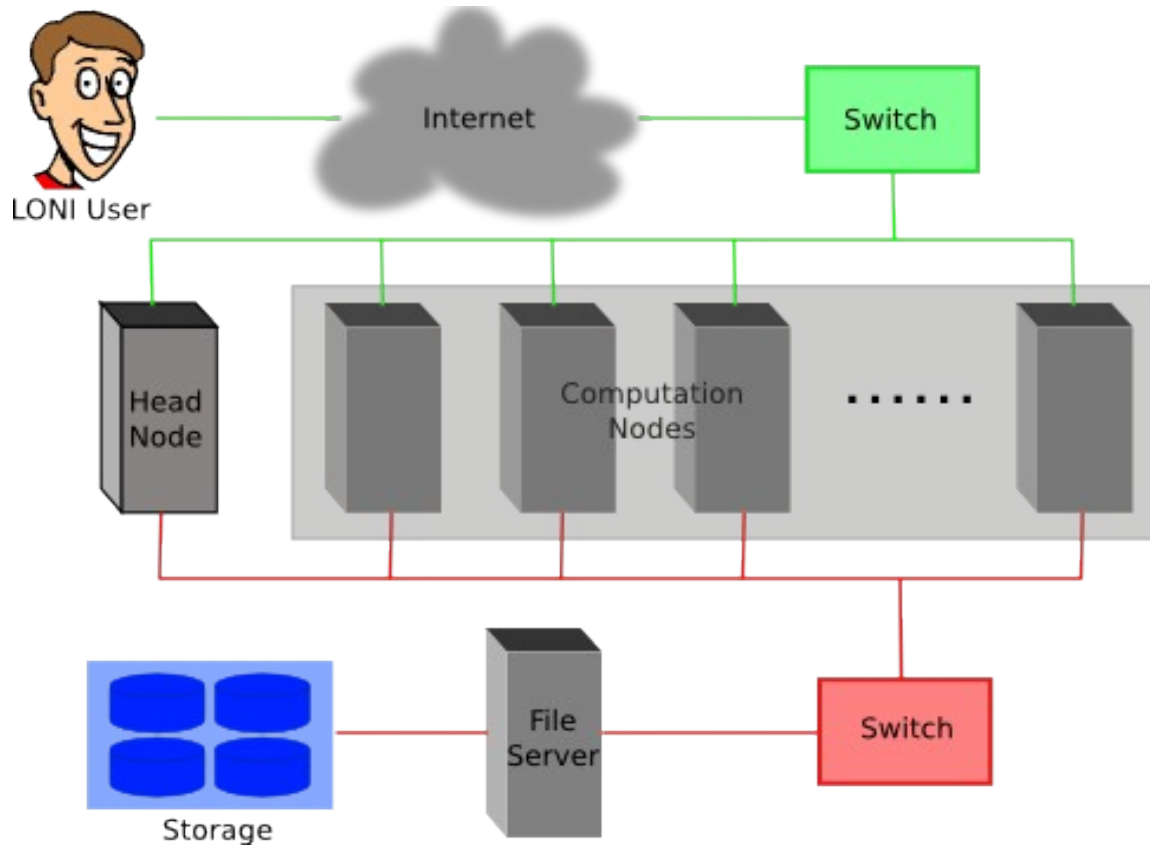
# Current deployment status - IBM AIX clusters

	Name	Peak TeraFLOPS	Location	Status	Login
LONI	Bluedawg	0.85	LaTech	Available	LONI
	Ducky	0.85	Tulane	Available	LONI
	Zeke	0.85	ULL	Available	LONI
	Neptune	0.85	UNO	Available	LONI
	Lacumba	0.85	Southern	Available	LONI

Manage your account:  
LONI <https://allocations.loni.org>



# 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 of which has **8 Intel Xeon cores @ 2.33 GHz, 8 GB RAM**
  - **192 TB storage**
- Other LONI Linux clusters
  - **128 nodes**: each of which has **4 Intel Xeons cores @ 2.33 GHz, 4 GB RAM**
  - **9 TB storage**
- LONI AIX clusters
  - **14 power5 nodes**, each of which has **8 IBM Power5 processors @ 1.9 GHz, 16 GB RAM**
  - **280 GB storage**





# Outline

- Hardware overview
- User 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](http://allocations.loni.org) and click on “profile”



# File Systems

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

- Tips

- Never let your job write output to your home directory
- 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



# Disk Quota

Cluster	Home		Work		Local scratch
	Access point	Quota	Access point	Quota	Access point
LONI Linux	/home/\$USER	5 GB	/work/\$USER	100 GB	/var/scratch
LONI AIX	/home/\$USER	500 MB	/work/default/\$USER	20 GB	/scratch/local

- 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





# Exercise 1

- Log in any LONI cluster
- Check your disk quota
  - Linux clusters: use “`showquota`” command
    - Your scratch directory will be created within an hour of the first login
  - AIX clusters: use “`quota`” command
- Locate the directory `/home/lyan1/traininglab/environment`
  - There are files that you will need for following exercises



# Setting up Software Environment

- Environment variables
  - PATH: where to look for executables
  - LD\_LIBRARY\_PATH: where to look for shared libraries
  - Other custom environment variables needed by some software packages
- **SOFTENV** is a software that is used to set up these environment variables on all the clusters
  - More convenient than setting numerous environment variables in `.bashrc` or `.cshrc`



# Listing All Packages

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

```
[lyan1@tezpur2 ~]$ softenv
```

```
...
```

```
These are the macros available:
```

```
* @default
* @globus-4.0          globus client
* @intel-compilers    compiler: 'Intel Compilers', version: Latest.
                      A pointer to the latest installed intel
                      compilers.
```

```
These are the keywords explicitly available:
```

```
+Mesa-6.4.2          No description yet for Mesa-6.4.2.
+R-2.8.0-gcc-3.4.6   application: 'R', version 2.8.0
+ansys-lsdyna-11.0   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/
```

Softenv key

```
...
```



# 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:

These are the keywords explicitly available:

+fftw-3.1.2-gnu	application: FFTW, version 3.1.2, binded with GNU compiler.
+fftw-3.1.2-intel10.1	application: FFTW, version 3.1.2, binded with Intel compiler v10.1.
+fftw-3.1.2-intel9.1	application: FFTW, version 3.1.2, binded with Intel compiler v9.1.

...



# 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
```



# 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
```



# 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  
-----
```





# 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`
- Set up your environment to use VISIT
  - Use `soft add +visit`
  - Or add “+visit” 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

Language	Linux clusters			AIX clusters
	Intel	GNU	PGI	XL compilers
Fortran	ifort	g77	pgf77,pgf95	xlf,xlf_r,xlf90,xlf90_r
C	icc	gcc	pgcc	xlc,xlc_r
C++	icpc	g++	pgCC	xlC,xlC_r

- 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



# Compilers for MPI Programs

Language	Linux clusters	AIX clusters
Fortran	mpif77,mpif90	mpxlf,mpxlf_r,mpxlf90,mpxlf90_r
C	mpicc	mpcc,mpcc_r
C++	mpiCC	mpCC,mpCC_r

- 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 on Linux clusters (1)

```
[lyan1@qb2 ~]$ ls -ld /usr/local/packages/mvapich*  
drwxr-xr-x 12 root root 4096 Oct 18 13:25 /usr/local/packages/mvapich-0.98-gcc  
drwxr-xr-x 12 root root 4096 Jan 23 11:35 /usr/local/packages/mvapich-0.98-intel10.1  
drwxr-xr-x 12 root root 4096 Oct 18 13:25 /usr/local/packages/mvapich-0.98-intel9.1  
drwxr-xr-x 12 root root 4096 Oct 18 13:25 /usr/local/packages/mvapich-0.98-intel9.1-LM  
drwxr-xr-x 12 root root 4096 Feb 12 10:27 /usr/local/packages/mvapich-0.98-pgi6.1  
drwxr-xr-x 12 root root 4096 Oct 18 13:25 /usr/local/packages/mvapich-0.98-pgi6.1-eric  
...  
drwxr-xr-x 10 root root 4096 Oct 18 13:25 /usr/local/packages/mvapich2-0.98-intel9.1  
drwxr-xr-x 11 root root 4096 Nov 9 16:31 /usr/local/packages/mvapich2-1.01-intel10.0  
drwxr-xr-x 9 root root 4096 Jan 25 09:54 /usr/local/packages/mvapich2-1.0.1-intel10.1  
drwxr-xr-x 11 root root 4096 Nov 8 13:10 /usr/local/packages/mvapich2-1.01-intel9.1
```

- 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 you code with the same version!!!
- Use the default version if possible



# MPI Compilers on Linux Clusters (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
    - Header files, libraries etc.
  - What they actually do can be reveal by the `-show` option

```
[lyan1@tezipur2 ~]$ 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
```





# 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
  - ...





# 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/environement`
  - Compile it with a serial compiler and see what happens
  - Compile it with an MPI compiler



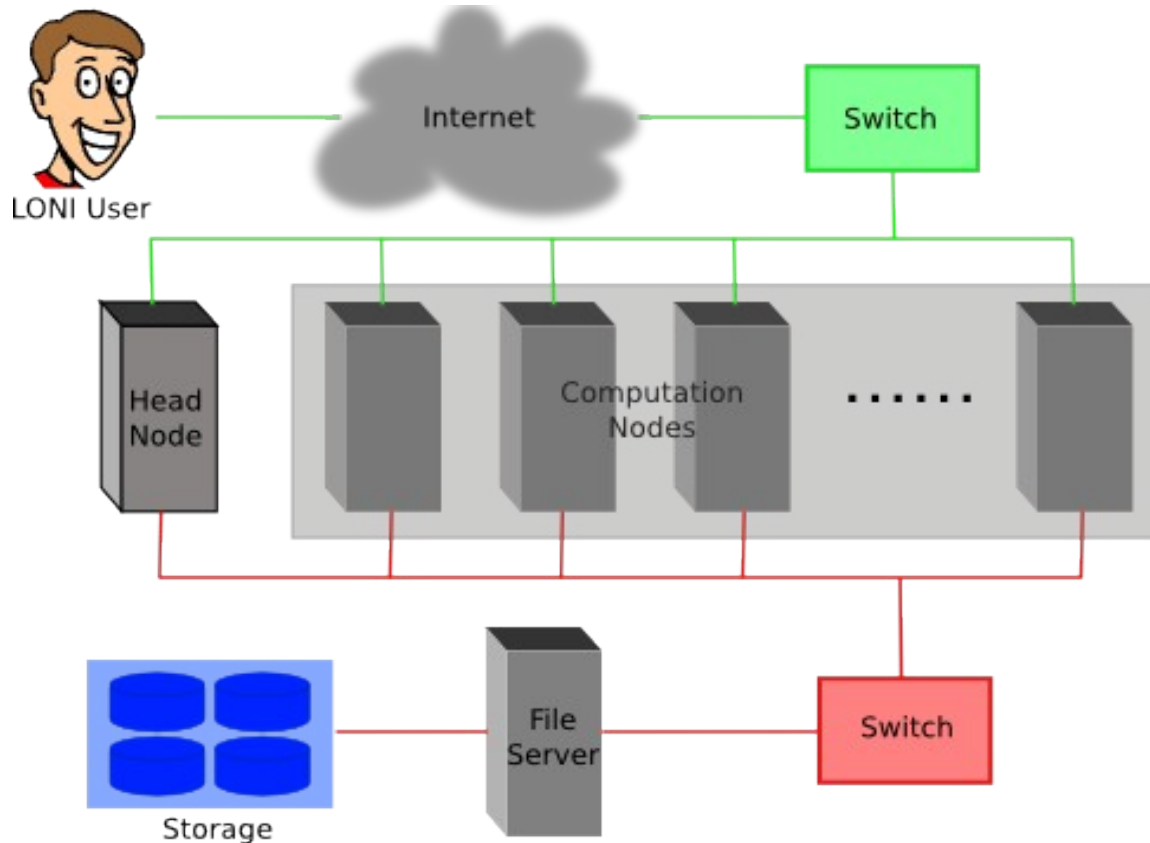


# Outline

- Hardware overview
- User environment
- Job management



# The Cluster Environment



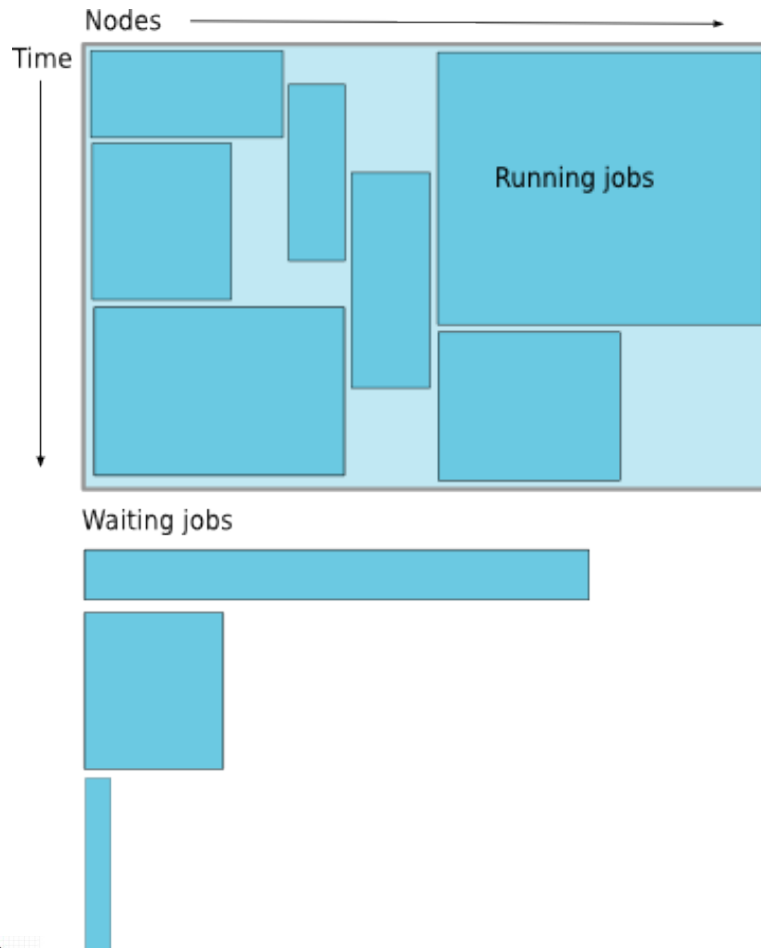
- Multiple compute nodes
- Multiple users
- Multiple jobs for each user

# 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**
- A job can be considered as 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



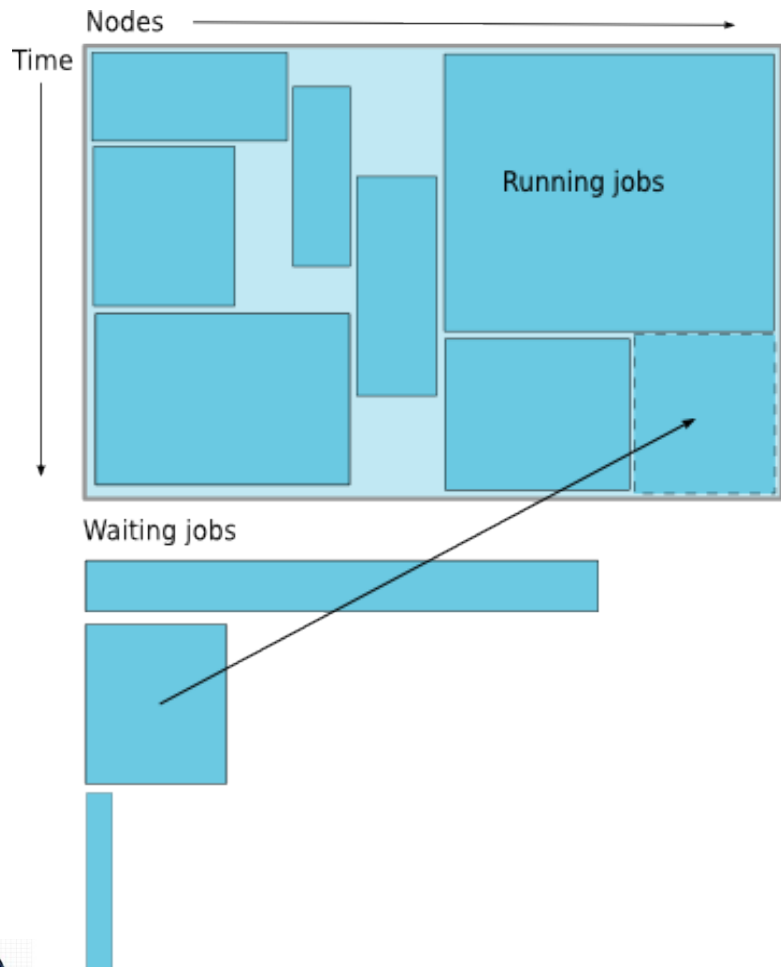
# A Simplified View of Job Scheduling



- Map jobs onto the node-time space
  - Assuming CPU time is the only resource
- Need to find a balance between
  - Honoring the order in which jobs are received
  - Maximizing resource utilization



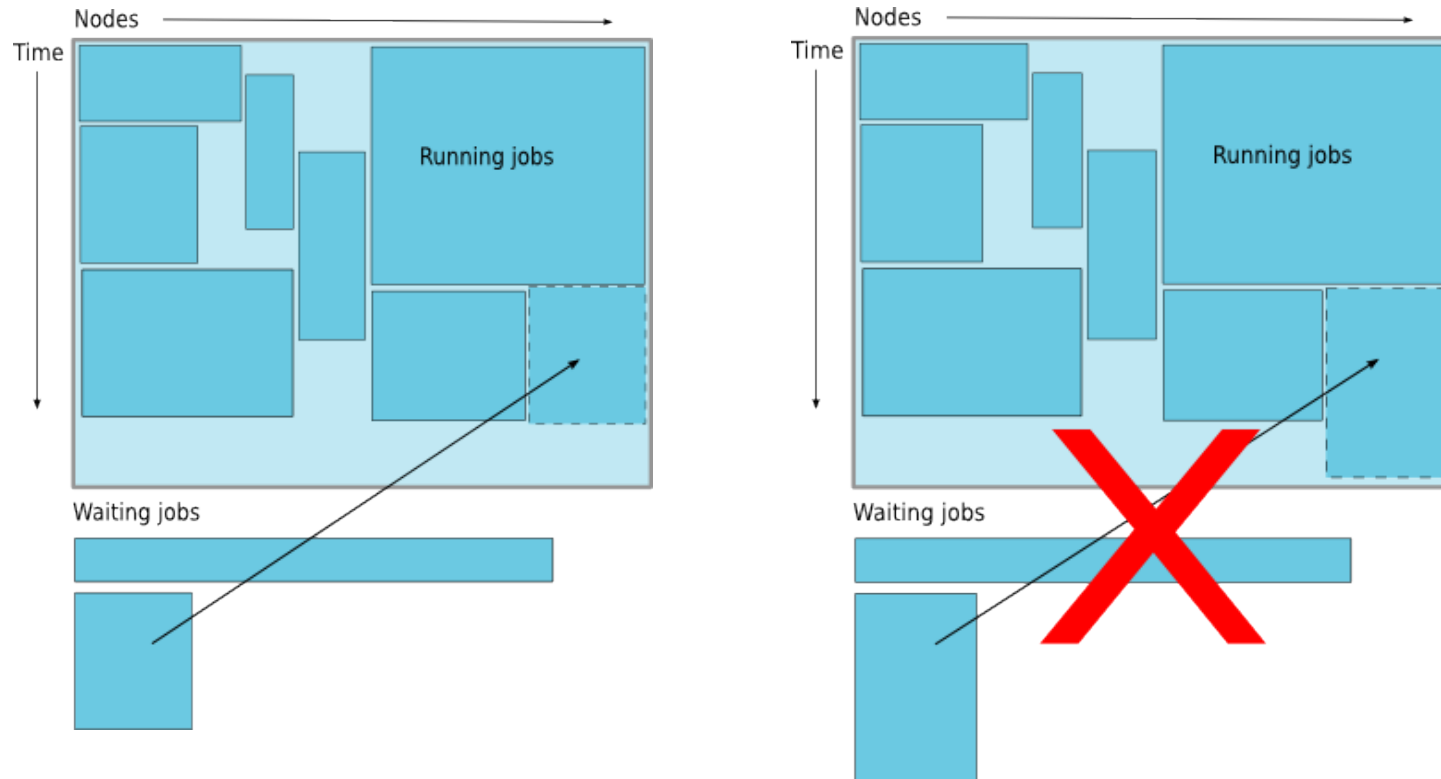
# Backfilling



- 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 HPC 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
  - ...



# Queue Characteristics – Queen Bee

Queue	Max Runtime	Total number of available nodes	Max running jobs per user	Max nodes per job	Use
Workq	2 days	530	8	128	Unpreemptable (default)
Checkpt		668		256	Preemptable jobs
Preempt		668	NA		Require permission
Priority		668	NA		Require permission



# Queue Characteristics – Other LONI Linux Clusters

Queue	Max Runtime	Total number of available nodes	Max running jobs per user	Max nodes per job	Use
Single	14 days	16	64	1	Single processor jobs
Workq	3 days	96	8	40	Unpreemptable (default)
Checkpt		128		64	Preemptable jobs
Preempt		64	NA		Require permission
Priority		64	NA		Require permission



# Queue Characteristics – LONI AIX Clusters

Queue	Max Runtime	Total number of available nodes	Max running jobs per user	Max nodes per job	Use
Single	14 days	1	8	1	Single processor jobs
Workq	5 days	8		8	Unpreemptable (default)
Checkpt		14		14	Preemptable jobs
Preempt		6	NA	Require permission	
Priority		6	NA	Require permission	



# 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`

```
lyan1@l2f1n03$ llclass
```







Name	MaxJobCPU d+hh:mm:ss	MaxProcCPU d+hh:mm:ss	Free Slots	Max Slots	Description
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
checkpoint	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








# Checking Loads on All LONI Clusters

- Check Loads on all LONI clusters at [docs.loni.org](http://docs.loni.org)
- Updated every 15 minutes

## Dell Linux Clusters

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

## IBM P5 Clusters

System Name	Nodes	SMP Size	Total CPUs	Memory/Node	TFLOPS	Work Disk	Location	Load	Running jobs	Queued jobs
Bluedawg	14	8	104	16 GB	0.851	270 GB	LaTech		16	3
Ducky	14	8	104	16 GB	0.851	270 GB	Tulane		7	0
Zeke	14	8	104	16 GB	0.851	270 GB	ULL		1	0
Neptune	14	8	104	16 GB	0.851	270 GB	UNO		9	8
LaCumba	14	8	104	16 GB	0.851	270 GB	SU		8	8



# 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 (**Don't run your test jobs on the head node!**)
- 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 be 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>`
- Add “-X” to enable X11 forwarding

- 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

<code>#!/bin/bash</code>	
<code>#PBS -l nodes=4:ppn=4</code>	Number of nodes and processor
<code>#PBS -l walltime=24:00:00</code>	Maximum wall time
<code>#PBS -N myjob</code>	Job name
<code>#PBS -o &lt;file name&gt;</code>	File name for standard output
<code>#PBS -e &lt;file name&gt;</code>	File name for standard error
<code>#PBS -q checkpt</code>	Queue name
<code>#PBS -A &lt;loni_allocation&gt;</code>	Allocation name
<code>#PBS -m e</code>	Send mail when job ends
<code>#PBS -M &lt;email address&gt;</code>	Send mail to this address
<code>&lt;shell commands&gt;</code>	
<code>mpirun -machinefile \$PBS_NODEFILE -np 16 &lt;path_to_executable&gt; &lt;options&gt;</code>	
<code>&lt;shell commands&gt;</code>	



# PBS Job Script – Serial Jobs

```
#!/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 Batch 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 = checkpoint
#@ 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



# 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
```

Id	Owner	Submitted	ST	PRI	Class	Running	On
12f1n03.3697.0	collin	1/22 16:59	R	50	single	12f1n14	
12f1n03.3730.0	jheiko	1/28 13:30	R	50	workq	12f1n10	
12f1n03.3726.0	collin	1/26 08:21	R	50	single	12f1n14	
12f1n03.3698.0	collin	1/22 17:00	R	50	single	12f1n14	
12f1n03.3727.0	collin	1/26 08:21	R	50	single	12f1n14	

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



# Job Monitoring – AIX Clusters

- **Command:** `showllstatus.py`
  - Show job status as well as node status

```
lyan1@peg304$ showllstatus.py
```

Node	Status	Load	Arch	Node	Status	Load	Arch
ben2	Idle	0.05	Power4	pen15	Run	8.04	Power5
ben3	Run	0.27	Power4	pen16	Idle	2.07	Power5
ian1	Idle	0.40	Power4	pen17	Down	0.01	Power5
pen01	Run	8.00	Power5	pen18	Idle	0.00	Power5
pen02	Busy	16.06	Power5	pen19	Busy	5.74	Power5
pen03	Busy	15.99	Power5	pen20	Idle	0.00	Power5

...

Step ID	Owner	Status	Class	Hosts	Queue	Date	Disp.	Date
ian1.77438.0	hypoxia	R	MP5L	4	02/10	10:26	02/10	10:26
ian1.77437.0	pradeep	R	SB4L	1	02/10	10:25	02/10	10:25
ian1.77431.0	eshi1362	R	MP5L	2	02/10	09:13	02/10	09:13
ian1.77419.0	jovi	R	MP5L	1	02/09	22:22	02/10	08:28
ian1.77418.0	jovi	R	MP5L	1	02/09	22:22	02/10	07:32
ian1.77417.0	jovi	R	MP5L	1	02/09	22:22	02/10	06:37

...



# 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`





# Where to Seek Help

- User's Guide
  - HPC: <http://www.hpc.lsu.edu/help>
  - LONI: [https://docs.loni.org/wiki/Main\\_Page](https://docs.loni.org/wiki/Main_Page)
- Contact us
  - Email ticket system: [syshelp@loni.org](mailto:syshelp@loni.org)
  - Telephone Help Desk: 225-578-0900
  - Walk-in consulting session at Middleton Library
    - Tuesdays and Thursdays only
  - Instant Messenger (AIM, Yahoo Messenger, Google Talk)
    - Add “lsuhpchelp”

