
# Intermediate HPC

## Introduction to Unix for HPC
## Raijin version

# What is HPC?

- HPC, or high-performance computing, refers to the application of supercomputers or clusters of computers to computational problems that typically arise through scientific inquiry
- HPC is useful when a computational problem:
  - **Is too large** to solve on a conventional laptop or workstation (because it requires too much memory or disk space) or
  - **Would take too long** (because the algorithm is complex, the dataset is large, or data access is slow) or
  - **Are too many** – High Throughput Computing

# Parallelism on HPC

- HPC systems often derive their computational power by exploiting parallelism
- Programs for HPC systems must be split up into many smaller "sub-programs" which can be executed in parallel on different processors
- HPC systems can offer parallelism at a much larger scale, with 100's or 1000's, or (soon) even millions of tasks running concurrently.
- Writing parallel software can be challenging, and many existing software packages do not already support parallelism & may require development.

**NOTE: Many tasks cannot be parallelised**

# Reasons to use HPC

- You have a program that can be recompiled or reconfigured to use optimized numerical libraries that are available on HPC systems but not on your own system.
- HPC applications are already installed on the HPC machines which is a non-trivial task
- You have a "parallel" problem, e.g. you have a single application that needs to be rerun many times with different parameters.
- You have an application that has already been designed with parallelism
- To make use of the large memory available
- Our facilities are reliable and regularly backed up

# When not to use HPC?

- You have a single threaded job which will only run one job at a time (typical of MatLab users)
- You rely on Databases
- You have a lot of data to transfer between your local machine and the HPC on a continuous basis (e.g. per job)
- You need to have a GUI to interact with your program

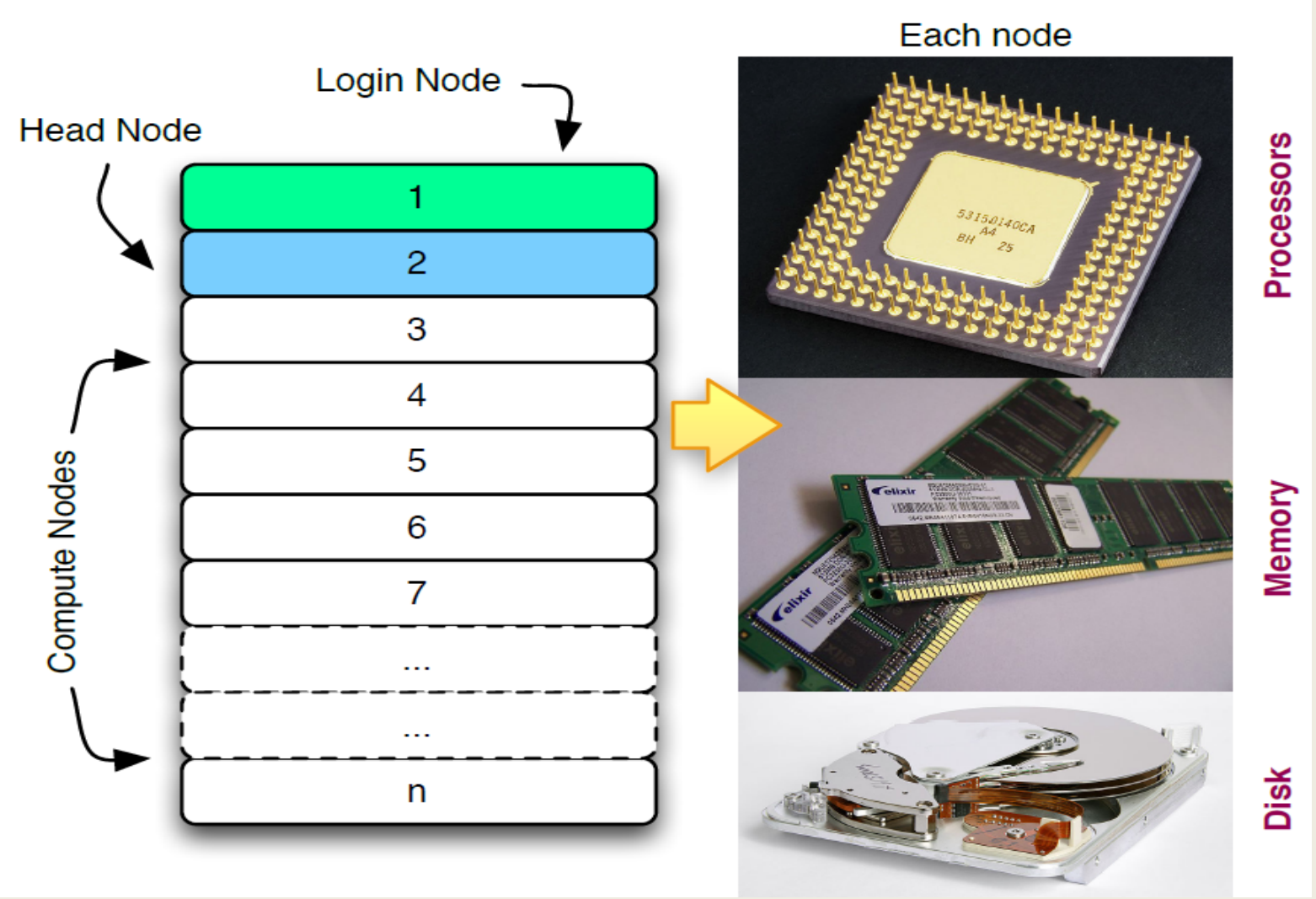
# HPC machines

| System | Memory Architecture | Cores | Nodes | Memory |
|---|---|---|---|---|
| **Raijin (NCI)** | **Distributed** | **57,472** | **3592** | **158TB** |

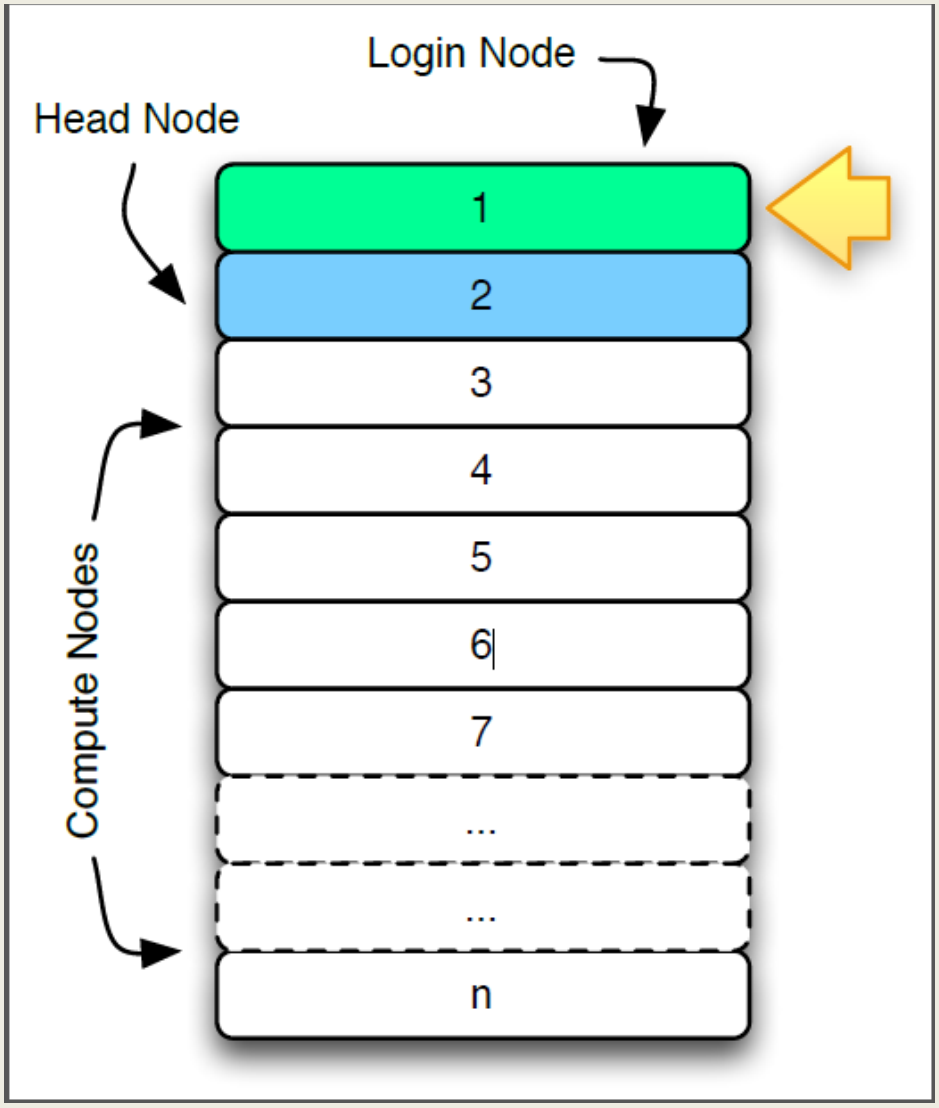
# The typical HPC workflow

- In HPC we talk about **jobs**, these are simply commands we wish to run and requests for resources (e.g. compute time, disk space, memory requirements, setup of s/w env's etc.)
- Generally time consuming & resource intensive.
- Jobs are typically run **non-interactively**,
- Can be run **interactively** for testing purposes
- We add our jobs to a **queue**.
- When machines have **free resources** jobs run
- Once jobs are complete, we can inspect their output.

# The HPC "Cluster"



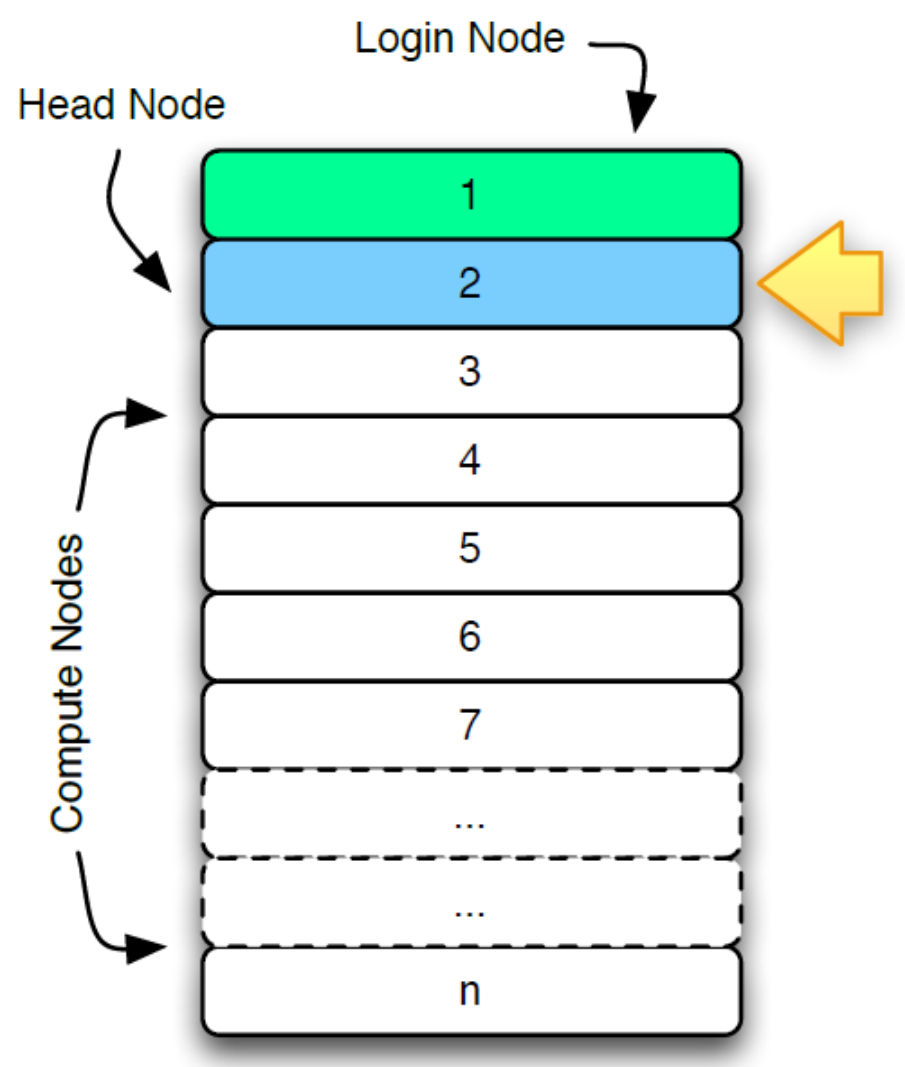
# The Login Node




**Login Node**

- Interactive programs
- SSH sessions
- Testing
- Compiling

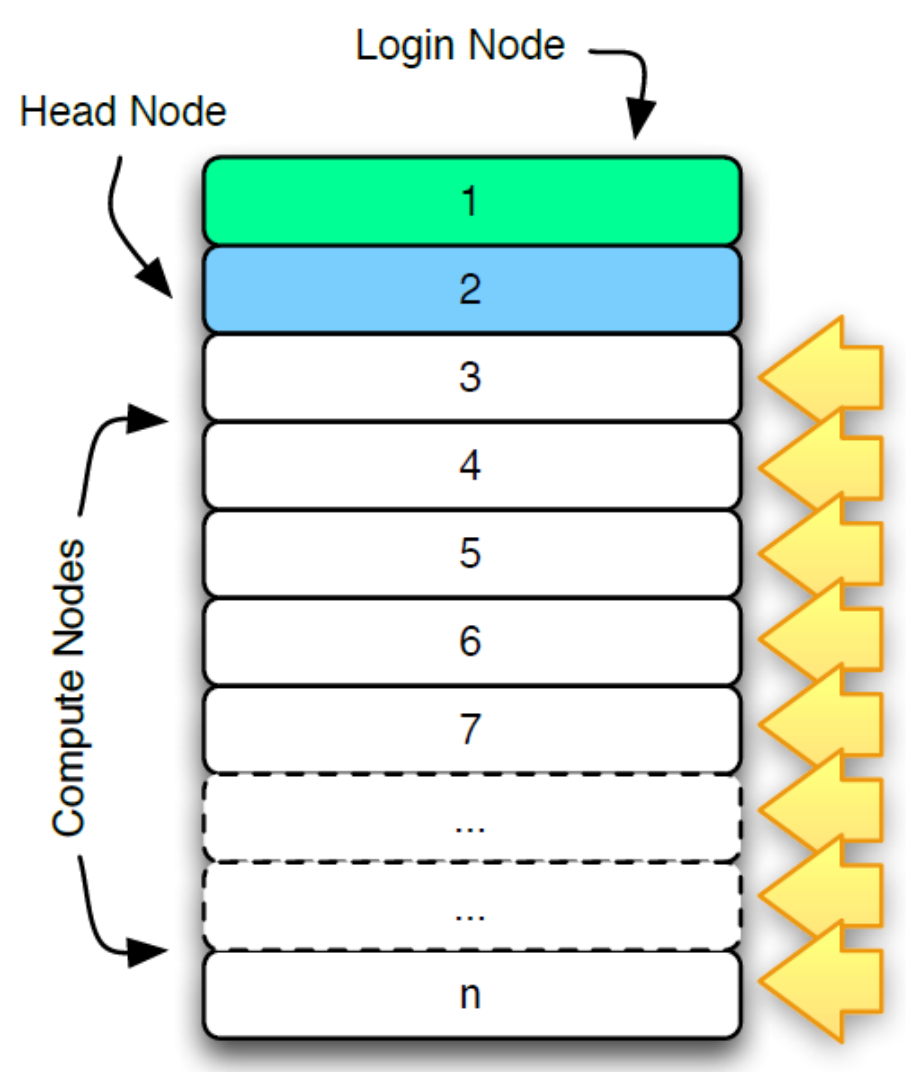
# The Head Node




**Head Node**

- Queuing jobs

# Compute Nodes




- These nodes run your jobs
- Managed by the **scheduler**
- Typically you won't interact with the nodes directly
- Some users may need to!

# Queuing Systems

- **Portable Batch System** (PBS) is the name of computer software that performs job scheduling. Its primary task is to allocate computational tasks, i.e., batch jobs, among the available computing resources.
- The following versions of PBS are currently available:
  - OpenPBS
  - TORQUE
  - PBS Professional (PBS Pro)
- Guide to PBS: http://hpc.sissa.it/pbs/pbs.html

# Queuing Systems cont.

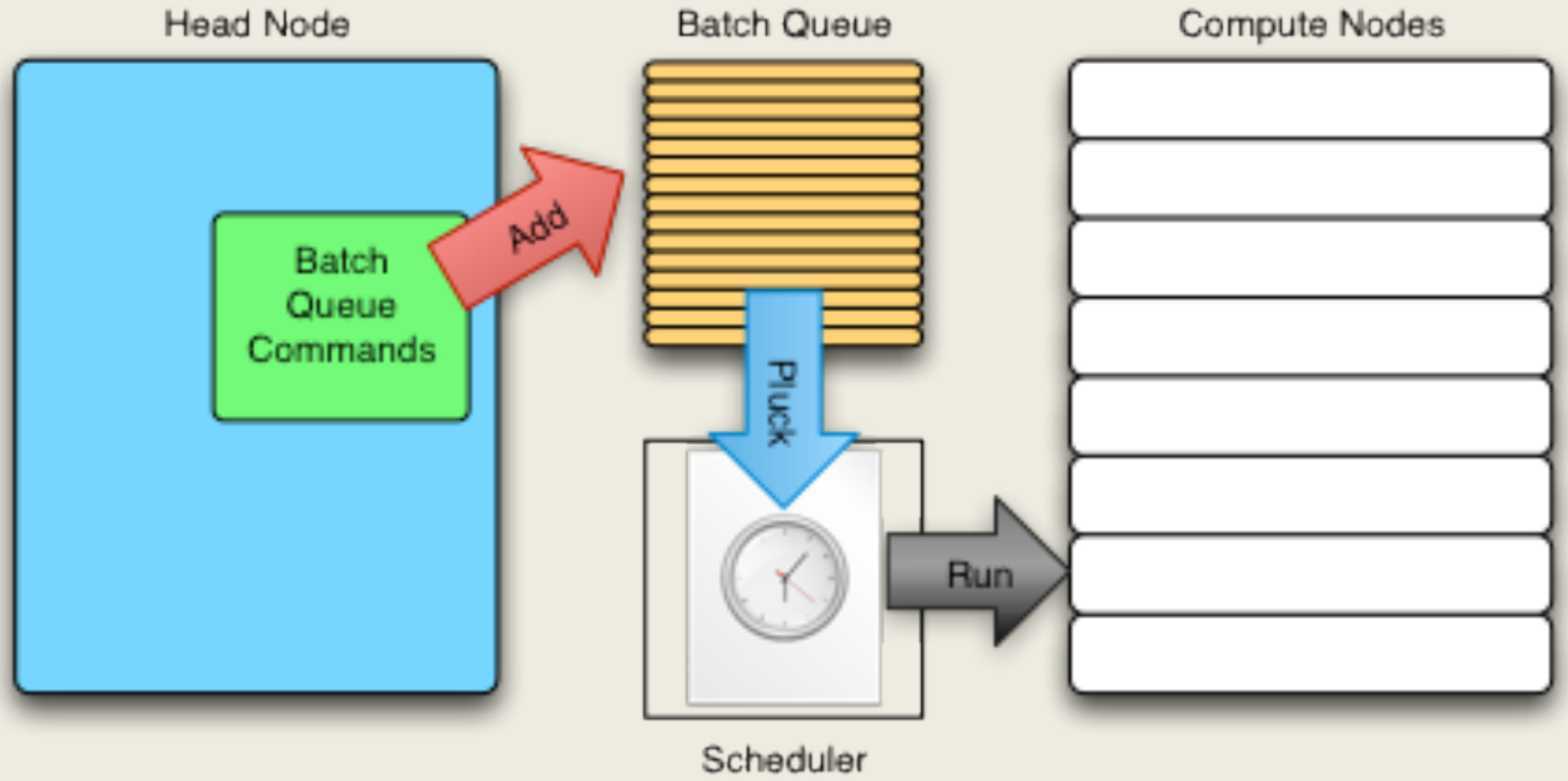
- Another popular batch system is **SLURM** (Simple Linux Utility for Resource Management )
  - Open source, fault-tolerant, and highly scalable cluster management and job scheduling system for large and small Linux clusters.
  - Very useful for use on clusters
  - Platform Tools used by IBM
  - Used by many supercomputers, e.g. TERA 100 at CEA (Europe's most powerful supercomp.)
- Many banks and commercial entities using batch systems

# PBS Pro

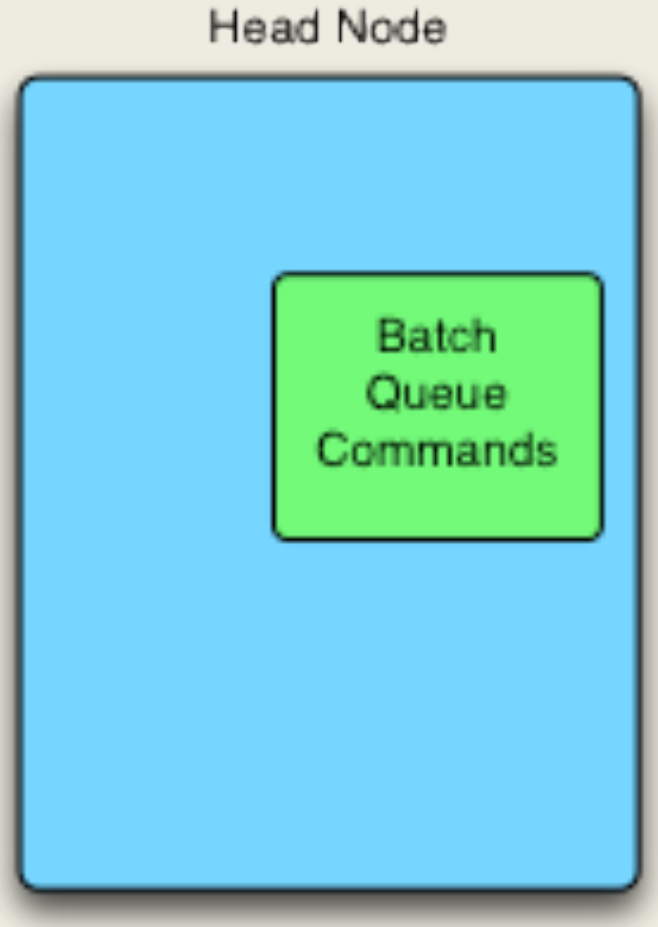
**PBS Pro** is the batch system used on Raijin.

| Batch System | PBS PRO |
| --- | --- |
| **Machines using** | Raijin |
| **Code Base** | PBS Professional |
| **Licence** | Altair Licence |

# The Batch Queuing System



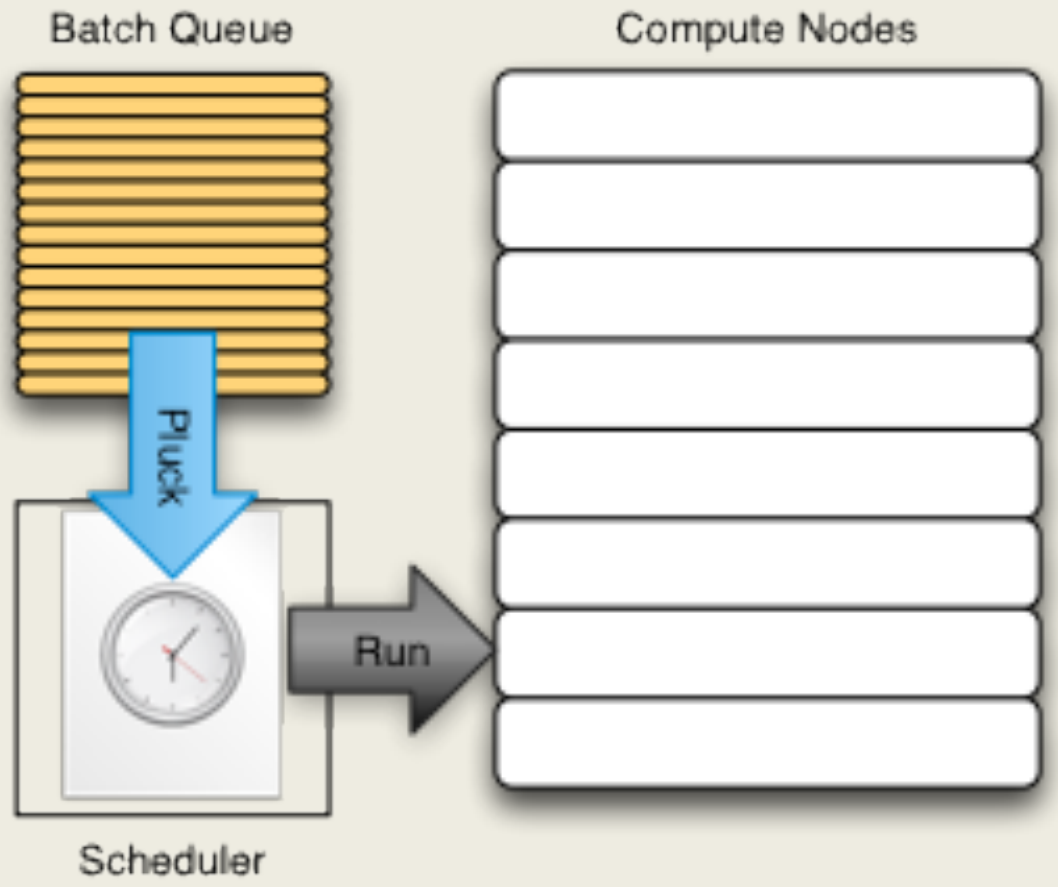
# Batch Queuing component




- The batch system is a normal program
- Lets you add and remove jobs from the queue and monitor the queue
- Script/command line driven

# The Scheduler component

- Allocates jobs to compute nodes
- Optimizes usage of resources
- "Optimize" can mean many things
- Non-trivial
- Never interact with directly



# PBS Pro Commands

In order to use the batch system productively, we need to know how to perform three actions:

- Add a job to the queue
- Remove a job from the queue
- See where our job is in the queue

| Command | Description |
|---|---|
| `qsub <job-script>` | Submit a job (add to queue)<br>Returns a <job-number> |
| `qdel <job-number>` | Delete job (remove from the queue) |
| `qstat <job-number>` | Monitor jobs |
| `qalter <job-number>` | Modifies the attributes of the job or jobs |

# Exercise 1

## Monitoring the queue with qstat

| Command | Description |
| --- | --- |
| `qstat -a` | List all jobs in the queue |
| `qstat -u <username>` | List all jobs of a particular user |
| `qstat -f <job-number>` | Show detailed information about a job |

# All about Modules

- **What do Modules do?**
  - Set up the environment for a software package
  - Adds paths to executables to $PATH
  - May change other shell variables, and/or load other modules.
  - Allow you to have different versions of the same software package, e.g. load the Intel compilers v23 with module load intel/12, or if you require an older version, you can load it with module load intel/9

# All about Modules (cont.)

**Things to know about modules**

- Only the pbs module is loaded when you login. You can change this by adding lines to your .bashrc file (BASH users) or similar for other Shells
- Modules exist for many packages but not for all!
- Some modules exclude each other, e.g.
  - You can't load Intel compilers v8 & v9. You can only load one.
  - You can only load 1 MPI (Intel or OpenMPI)

# Module Commands

| Command | Description |
| --- | --- |
| **`module avail`** | Will list all available module files in the current MODULEPATH |
| **`module load/unload`** | Will load/unload a modulefile into the shell environment<br>e.g. **`module load mpt/2.06`** |
| **`module list`** | List all loaded modules |
| **`module show [modulefile]`** | Will show information about the modulefile |

# Add a job to the queue

- To add a job to the queue, we write a **job script**. A job script is a simply a script.
- The # symbol signifies a comment for the Shell
- It has some special comments that pass info to PBS Pro.
- **#PBS** is a keyword for PBS and specifies that this line is for PBS. The Shell will ignore it
- When we want to queue the job, we pass its filename as a parameter to **qsub**, e.g.
  - **qsub <job-script>**
- The batch queuing system will return a number that uniquely identifies the job.

# Useful Environment Variables

- These are available in the context of your job script.

| Command | Description |
| --- | --- |
| `PBS_O_WORKDIR` | The directory the job was submitted from |
| `PBS_JOBID` | The job number given when the job was submitted |

# A sample PBS job script for Orange

Note the "-l" below is a lowercase letter "L" not a one "1"!

```
#!/bin/bash
# * Specify your project code
#PBS -P do18
# Request resources
# * 10 minutes wall time to run
#PBS -l walltime=00:10:00
# * 1 node, 1 processor
# * 100 megabytes physical memory allocated to job
#PBS -l ncpus=1
#PBS -l mem=100mb
# Specify the express queue
#PBS -q express
cd $PBS_O_WORKDIR
# Specify the job to be done
date
sleep 60
date
```

# You’ve got mail!

```
# Set email address
#PBS -M someone@example.com
# Send an email when jobs
# begins (b), gets aborted (a)
# and ends (e)
#PBS -m abe
```

# Exercise 2

## Create a script and submit a very simple job

| Command | Description |
|---|---|
| **`#PBS`** | **`#PBS`** is a keyword for PBS and specifies that this line is for PBS. The Shell will ignore it |
| **`#`** | Signifies a comment for the Shell, e.g. **`# Next line will create a job`** |
| **`qsub`** | Submit a job to PBS |
| **`qstat`** | List jobs in the queue |
| **`cat`** *<filename>* | Print a file to the terminal (catenate) |
| **`less`** *<filename>* | Like **`cat`**, but less at a time |
| **`nano`** *<filename>* | Will open file *<filename>* in a text file editor |
| **`Sample PBS Script`** | *see exercises document* |

# Exercise 3

## Create another script and submit a more realistic sample job

| Command | Description |
|---|---|
| **`#PBS`** | **`#PBS`** is a keyword for PBS and specifies that this line is for PBS. The Shell will ignore it |
| **`#`** | Signifies a comment for the Shell, e.g. **`# Next line will create a job`** |
| **`qsub`** | Submit a job to PBS |
| **`qstat`** | List jobs in the queue |
| **`nano`** *<filename>* | Will open file *<filename>* in a text file editor |
| **`module load`** <module_name> | Loads a software module on the HPC machine |

# Job limits on Raijin

- 48hours of **walltime**
- 32GB, 64GB, 128GB, 256GB nodes are standard (scheduler will decide)
- 3 x 1TB available via hugemem queue
- 3 x 3TB coming soon
- **NOTE**: If you grab a node with 64GB, you can effectively use about 60GB as the OS uses memory

# Priorities of Jobs

In order of importance, jobs are prioritised in this order:

1. Resources available to the project
2. Walltime

# Raijin queues

| Queue | Charge weight | Comments |
|---|---|---|
| **normal** | 1.00 | Normal queue |
| **express** | 3.00 | Express queue. Jobs are submitted faster. |
| **normalbw** | 1.25 | Queue using Broadwell nodes (can be cost effective) |
| **expressbw** | 3.75 | Express queue using Broadwell nodes |
| **knl** | 0.25 | |
| **gpu** | 3.00 | |
| **gpupascal** | 4.00 | |
| **hugemem** | 1.25 | For access to the 3 x 1TB and 3 x 3TB nodes |

# NCI Facilities

| Disk Type | Disk Usage |
| --- | --- |
| **NCI Raijin Sandy Bridge** | • **2 sockets with 8-core CPUs = 16 cores**<br>• 57,472 cores in the compute nodes<br>• Approximately 160 TBytes of main memory;<br>• Infiniband **FDR** (five data rate) interconnect |
| **NCI Raijin Broadwell** | • **2 sockets with 14-core CPUs = 28 cores**<br>• ~23,000 cores in the compute nodes<br>• Infiniband **EDR** (eight data rate) interconnect |
| **NCI Raijin Lustre Filesystem** | • Approximately 42 PBytes of usable fast file system |

# NCI Facilities

| Raijin (Sandy Bridge) |
| --- |
| 96.5% of Nodes have 32GB/node |
| 3.2% of Nodes have 64GB/node |
| 0.3% of Nodes have 128GB/node |
| *Plus* 3 nodes with 1TB/node |
| *Plus* 3 nodes with 3TB/node (coming soon) |

# Software on Raijin

| Area | Software |
|---|---|
| Computational Chemistry | ABINIT, Amber, CPMD*, GULP*, NAMD*, Molpro etc. |
| Bioinformatics | AbySS, BEAST, BIOPERL, Cufflinks, MAW, etc. |
| Math Libraries | ARPACK, BLACS, Boost, FFTW, GSL, MKL, Tao |
| Statistics & Maths Env's | Maple*, Mathematica*, MatLab*, Octave*, R, Stata* |

- Asterisked items indicates that discussion with NCI facility staff is required before use (Licensing issues) https://opus.nci.org.au/display/Help/Application+Software

# Disk Partitions

| /g/data1, 2, 3 | |
|---|---|
| Mounted under: | `/g/data[1,2,3]` |
| Size: | Depending on project allocation |
| Backed up: | **No** |
| Life time: | Permanent |

| Massdata | |
|---|---|
| Mounted under: | `/mdss` |
| Size: | Depending on project allocation. Intended to be a backup. |
| Life time: | Permanent |

# Disk Partitions

| /home | |
|---|---|
| Mounted under: | `/home/{username}` |
| Size: | 2GB |
| Backed up: | Yes |
| Life time: | Permanent |

| /short | |
|---|---|
| Mounted under: | `/short/{project_code}` |
| Size: | Depending on the project allocation |
| Backed up: | **No** |
| Life time: | **60 days** |

# Disk Partitions

You can find out more about the partitions on the HPC machine using the **df** command.

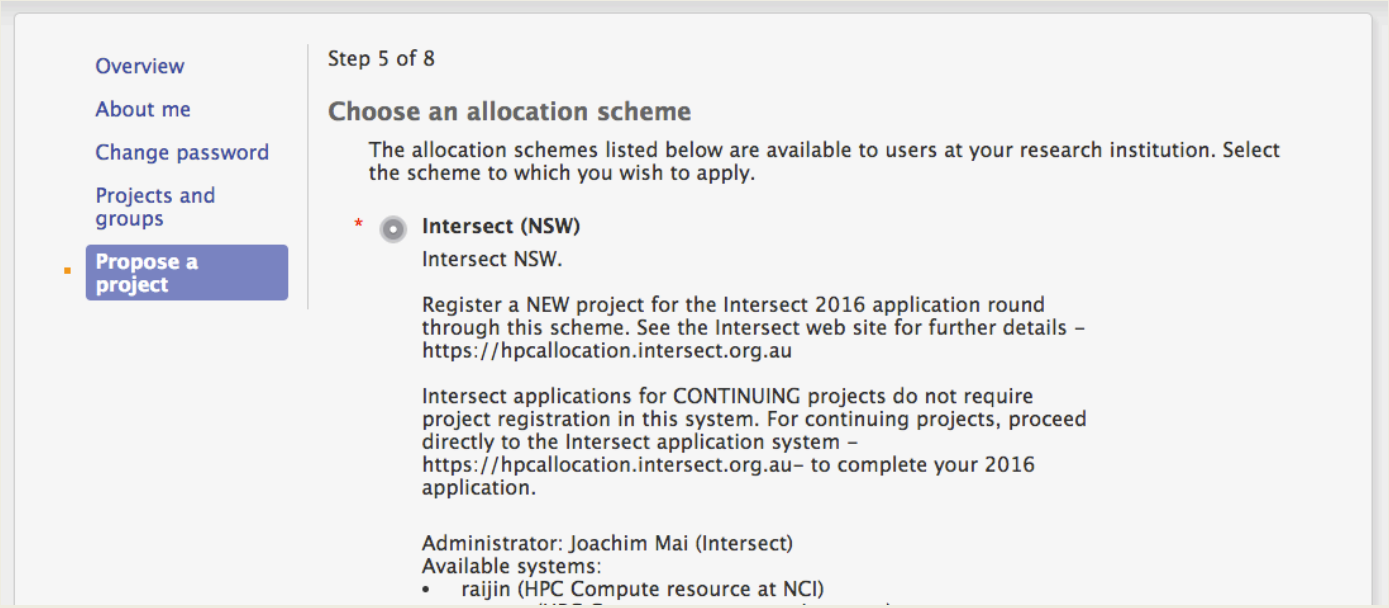
| Command | Description |
| --- | --- |
| `df -h` | Show disk free space for all partitions in human readable format |

You can find out more about current disk usage, using the **du** command.

| Command | Description |
| --- | --- |
| `du -hs .` | Show disk usage of current directory in human readable format |

# To Apply

- Create an ID in NCI Mancini
- Apply for a new project (*propose a project*)
- Select Intersect under allocation scheme



# Applications continued

- If you have any problems requesting resources, email help@intersect.org.au who can advise you
- You can also add accounts to an existing project via Mancini
- You can also add various software groups (e.g., MATLAB) via Mancini

# Conclusion

- In this course we have covered
  - the basics of the Unix command line
  - transferring data onto Raijin
  - how to queue a job on Raijin and get the output

# Thanks for attending!

Please complete our **course survey** at:

https://goo.gl/EbC9eO <- Capital 'Oh'

Any **further questions**, contact us at

training@intersect.org.au

INTERSECT