# Introduction to HPC Workshop

Centre for eResearch
(eresearch@nesi.org.nz)

# Outline

# About Us

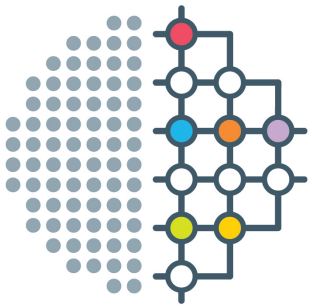
## CeR: Centre for eResearch

- Part of the University of Auckland
- User support and system maintenance

## NeSI : New Zealand eScience Infrastructure

- **NeSI** provides
  - high performance computing
  - a national data storage and sharing service
  - expert support and knowledge sharing
  - single-sign on across the NZ research sector

# About Us



NeSI
New Zealand eScience Infrastructure

THE UNIVERSITY OF AUCKLAND
NEW ZEALAND
Te Whare Wānanga o Tāmaki Makaurau

UC
UNIVERSITY OF CANTERBURY

NIWA
Taihoro Nukurangi

ministry of science + innovation
TE PŪNAHA HIHIRI O AOTEAROA

Manaaki Whenua
Landcare Research

UNIVERSITY of OTAGO
Te Whare Wānanga o Otāgo
NEW ZEALAND

# About Us

## Our Team

- We support researchers to get the most out of our platforms and services.
- As a team, we have a lot of experience in HPC that spans many science domains.
- Collaboratively enhance the performance of research software codes.
  - Troubleshoot memory and other or I/O bottlenecks.
  - Connect researchers and scientific software experts.
  - The team is available to support researchers across any research institution in New Zealand.

# About Us

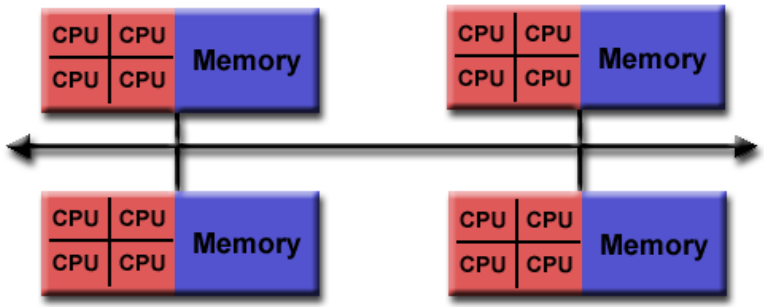
## Support

- Email `eresearch@nesi.org.nz`
- Creates a support 'ticket' where we can track the history of your request
- You can also arrange to meet us to discuss any issues

# Key Concepts

## What is a cluster

- A cluster is a network of computers, sometimes called nodes or hosts.
- Each node has several CPUs (or cores).
- A CPU does the computing.
- If your application uses more than one CPU, it can run faster on our cluster.



# Key Concepts

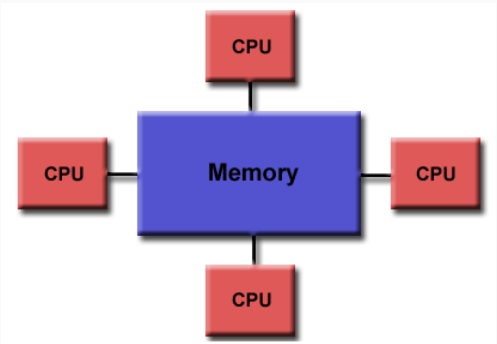
## Parallel Programming

- There are several ways to make a program use more CPUs and hence run faster.
- Many scientific software applications are written to take advantage of multiple CPUs in some way. But often this must be specifically requested by the user at the time he runs the program, rather than happening automagically.
- We can help you improve the performance of your code or make better use of your application.

# Key Concepts

## Shared Memory

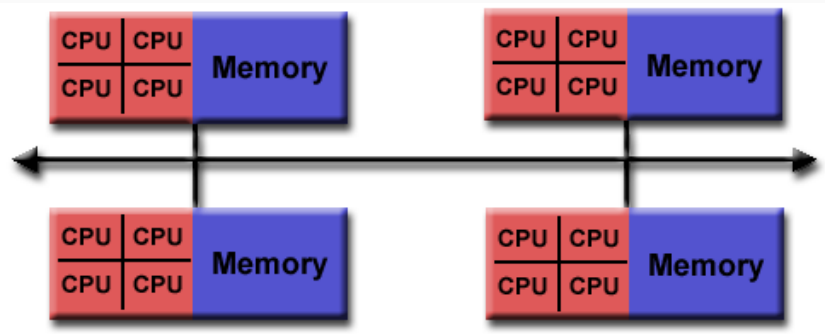
- Symmetric multiprocessing (SMP): two or more identical processors are connected to the same main memory.
- The program can divide tasks up between several threads. Each thread has access to all the program's data.
- There are different frameworks for utilizing SMP capabilities.



# Key Concepts

## Distributed Memory

- Multiple-processor computer system in which each process has its own private memory.
- Computational tasks can only operate on local data.
- If remote data is required, the computational task must communicate with one or more remote processors.
- The most popular parallel programming paradigm is MPI (Message Passing Interface).



# Our Facilities

## NeSI Facilities

NeSI provides several HPC architectures and solutions to cater for various needs:

- BlueGene/P
- Power6 and Power7
- Intel Westmere
- Intel SandyBridge
- Intel IvyBridge

- Kepler and Fermi GPU servers
- Intel Xeon Phi Co-Processor

# Our Facilities

## NeSI Facilities

- Many (though not all) supported applications can run on multiple NeSI architectures.
- We can install and test an application on all supported NeSI architectures and find the most suitable environment for your case.
- See the NeSI website for facility specifications and application details.

# Our Facilities

## Pan, the NeSI CeR Supercomputing Centre

- Funded by the **University of Auckland**, **Landcare Research** and the **University of Otago** with co-investment from the NZ Government through **NeSI**.
- Currently have around 6000 Intel CPUs across about 400 hosts.
- About 50 TB of memory.
- Shared storage of 400 TB with a 40 Gbit/s InfiniBand network.
- Runs Red Hat Enterprise Linux 6.3 as the operating system.

# Our Facilities

## NeSI Pan Cluster

| Architecture | Westmere | SandyBridge | IvyBridge | LargeMem |
|---|---|---|---|---|
| Clock Speed | 2.8 GHz | 2.7 GHz | 2.7 GHz | 2.4 GHz |
| CPUs/socket | 6 | 8 | 12 | 10 |
| CPUs/node | 12 | 16 | 24 | 40 |
| Mem/node | 96 GB | 128 GB | 128/256 GB | 512 GB |
| # nodes | 76 | 194 | 42 | 4 |

We also have GPUs and Intel Xeon Phi's available. More information on the documentation page under *Available hardware.*

# What to expect

## Suitable work

- Problems that can be solved with parallel processing.
- Problems that consume large amounts of memory.
- Problems that render your desktop useless for long periods of time.

## Less suited

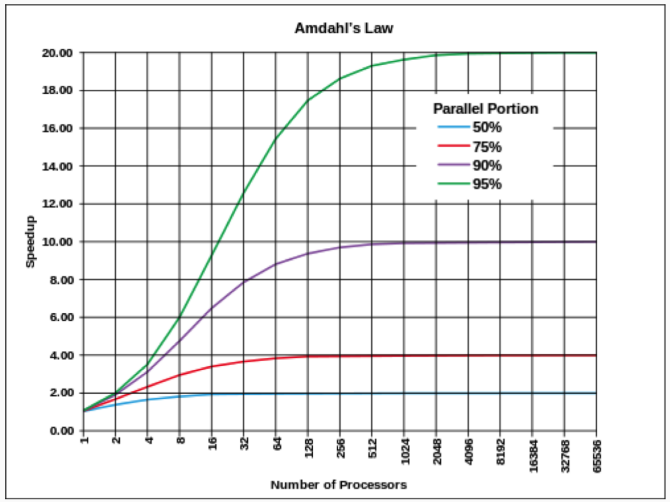
- Windows-only software $\mapsto$ Aspirational Research Virtual Machine Farm.
- Interactive software, e.g. GUI, only available for development (in general).

# What to expect

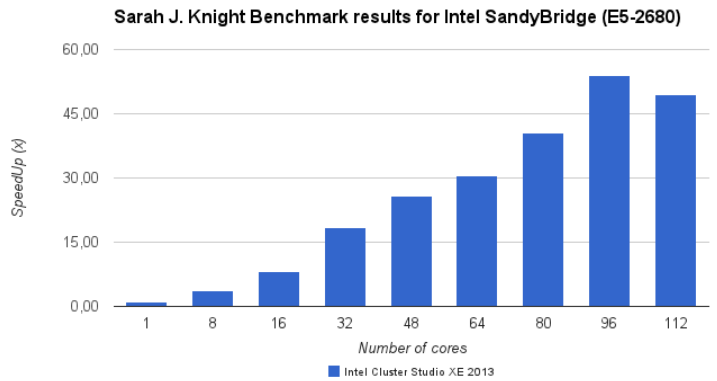
## Scaling behaviour

- Some problems are "embarrassingly parallel", i.e. it is trivial to divide the problem and solve independently.
  For instance, you could run the same simulation with 1000 different initial conditions.
- Approximately linear speedup.
- Other problems have dependencies, they cannot be separated e.g. simulating the weather.
- Speedup depends what % of the program runtime can be parallelised.

# Amdahl's Law




# Real Case: more CPUs ≠ more speed

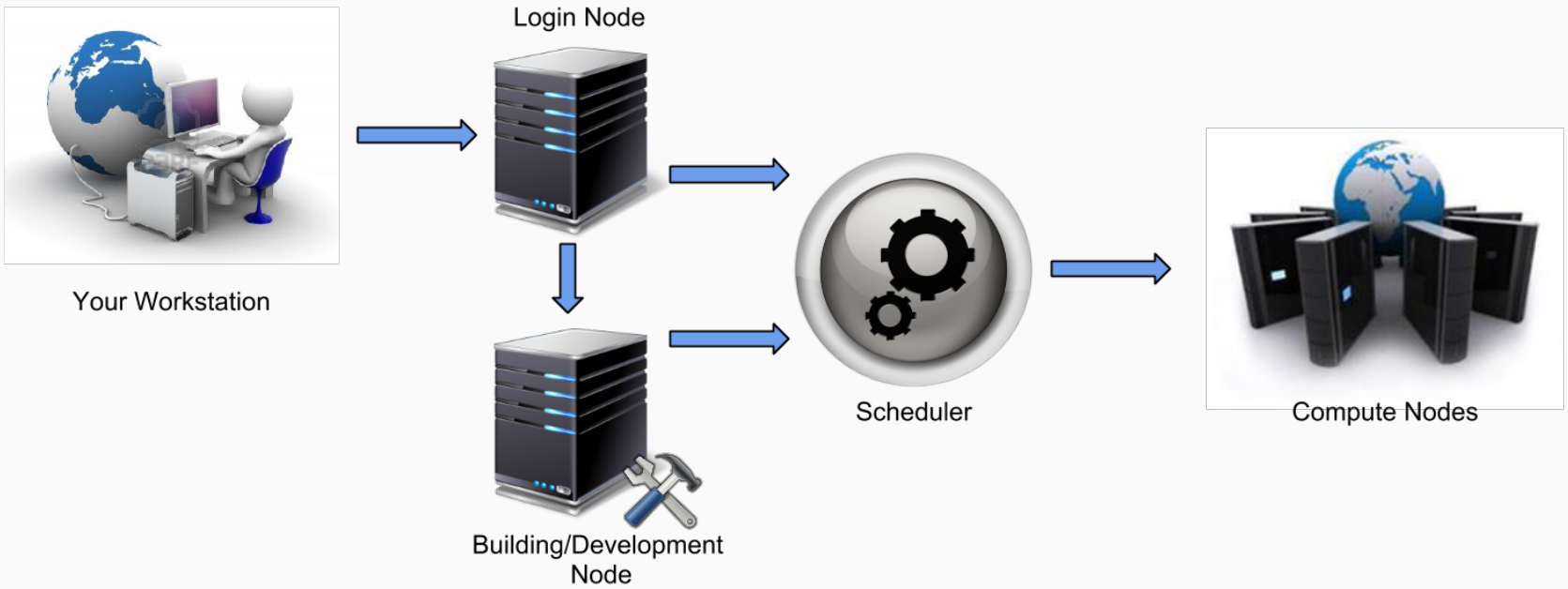


## Parallel execution time

- Single CPU computation time: computation only.
- Parallel computation time: computation + communication + waiting.
- For example:
  - Writing results (to one file) is often a bottleneck.
  - Small problem on many cores: communication costs will dominate.
  - Unbalanced load: one slow CPU will hold up all the others.
- Conclusion: Test which number of CPUs is best suited for your problem.

# General overview

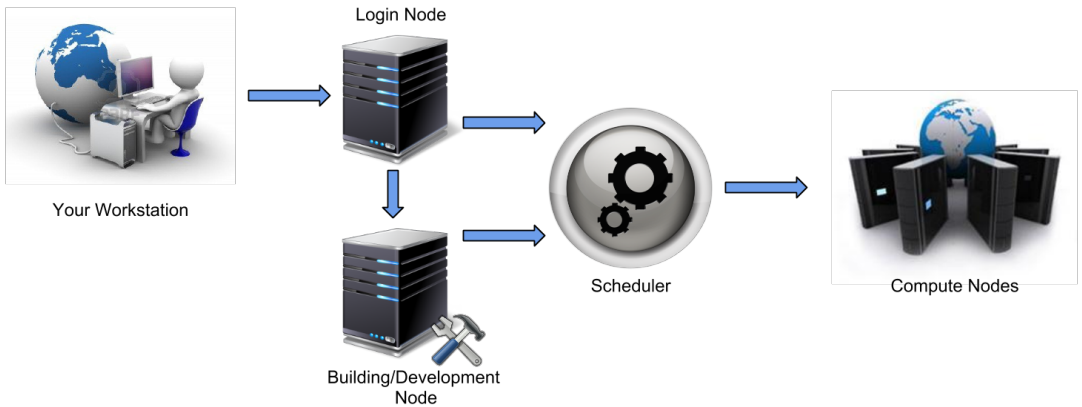
## Using the cluster



# Using the Cluster

## Overview

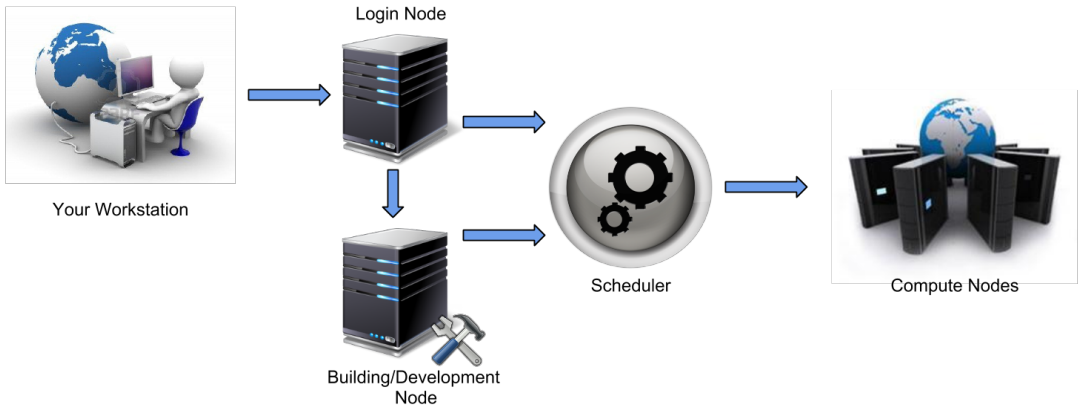
- The cluster is a shared resource and work must be scheduled.
- Jobs are queued and are executed on the compute nodes.
- The login node is not for running jobs, it is only for file management and job submission.



# Using the Cluster

## Compiling and Testing Software

- In each NeSI facility you will find building/development nodes.
- We have the most up-to-date development tools ready to use.
- You can build and test your software and then submit a job.



# Using the Cluster

## Connection via SSH

There is software available for each desktop operating system that implements the Secure Shell (SSH) protocol:

- Windows: MobaXterm (third-party, not included with the OS)
- Mac OS X: Terminal (shipped with the OS), iTerm2 (third-party, not included)
- Linux: Konsole, Gnome Terminal, Yakuake

Whichever terminal you use, you will need to run a command like: ssh `jbon007@login.uoa.nesi.org.nz`

# Using the Cluster

## Remote File System Access

In order to access the file system (/home) remotely from your machine, we recommend:

- **Windows** (mobaxterm): mobaxterm
- **MacOSX** (SSHFS): `http://code.google.com/p/macfuse/`
- **Linux** (SSHFS): `http://fuse.sourceforge.net/sshfs.html`
- **Linux** (scp): `https://linuxacademy.com/blog/linux/ssh-and-scp-howto-tips-tricks/`
- **KDE** (Konqueror): type fish://user@host:port
- **Gnome** (Nautilus): type sftp://user@host:port

# Using the Cluster

## Data

- Upload input data to the login node for use on the cluster.
- Download results from the login node to your local drive.
- Your home directory has a small quota. Project directories are significantly larger.
- Things do go wrong. Keep your own backups of anything important.
- For long-term storage and backups, consult your institution's IT department.
- Files on the login node are shared across the build and compute nodes.

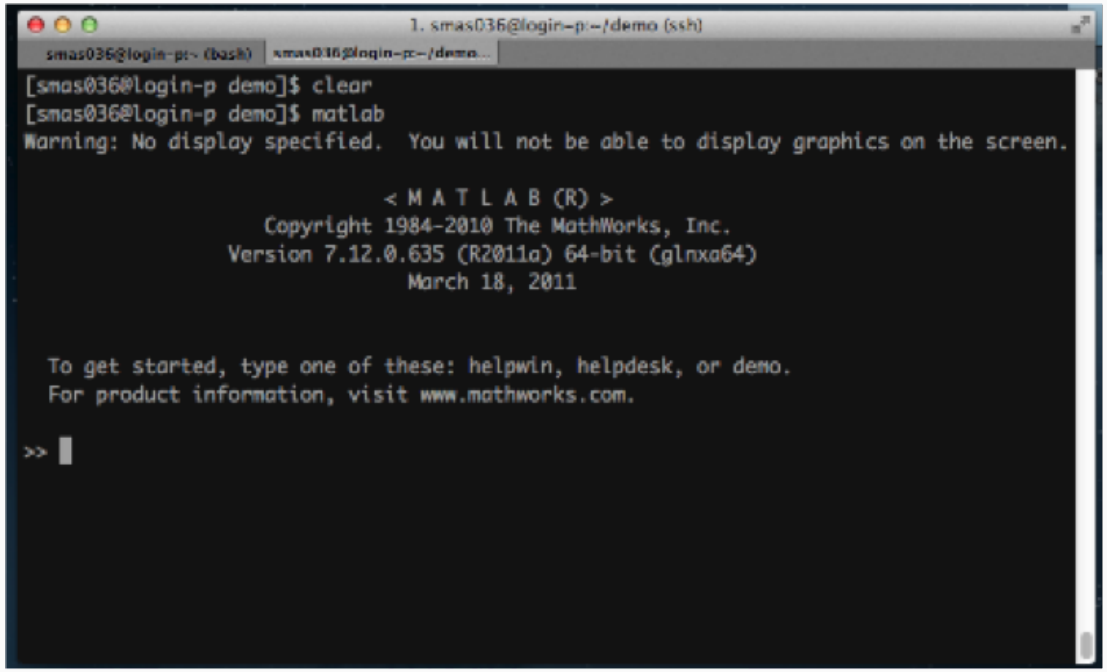
# Submitting a Job

## Basic Job Properties

- **Name**: For easily identifying the job (in the queue) and its output files.
- **Walltime**: How long can the job run for? The job will be killed if it runs out of time.
- **Memory**: How much to use? The job will die if it needs more memory than you allow.
- **CPUs**: How many to use? Some programs try to use more CPUs than they are allocated, e.g. MATLAB.
- **Account information**: Especially important for access to funded research allocations
- **Emails**: Whom, and in what circumstances, the scheduler will notify about changes to the job status.

# Submitting a Job

## Outputs




Jobs have no interactive interface, but write to files. Text written to the command line output and error channels will also be collected in files. Limited graphical tools are available on the login and build/development nodes.

# Submitting a Job

## Outputs

- Information output while the job runs is written to a text file
- Standard output and standard error are written to files named after the job, unless you specify different names
- These should have unique names for a given job directory (see job name)
- Other files produced during the job will keep their expected names, e.g. output data
- When your job fails, first look at the output and error files for clues

# Submitting a Job

## Environment Modules

- Modules are a convenient way to provide access to applications on the cluster
- They prepare the environment you need to run the application
- Some useful commands:
  - **module avail** - lists available modules.
  - **module show module_name** - displays full information about the module with name *module_name*.
  - **module load module_name** - loads the module with name *module_name* and its dependencies.
  - **module unload module_name** - unload the module with name *module_name* and its dependencies.
  - **module list** - list all modules currently loaded.

# Submitting a Job

## Quick introduction to Slurm

- You need to access the login node and work from a terminal.
- Requires basic knowledge of the Linux command line:
  - How to navigate the file system and edit text files.
  - Shell scripting is very useful for automation.
  - Tutorials available online at Software Carpentry – computing basics aimed at researchers.

# Submitting a job with SLURM: example job file

## Job Description Example: Serial

```
#!/bin/bash
#SBATCH -J MySerialJob
#SBATCH -A uoa99999          # Project Account
#SBATCH --time=01:00:00      # Walltime
#SBATCH --mem-per-cpu=4GB  # Memory per CPU

srun cat ~/inputfile.txt
```

Also see `https://wiki.auckland.ac.nz/display/CER/Slurm+User+Guide`

# Submitting a job with Slurm: example MPI job file

## SLURM job Description Example: MPI

```
#!/bin/bash
#SBATCH -J MyMPIJob
#SBATCH -A uoa99999          # Project Account
#SBATCH --time=01:00:00      # Walltime
#SBATCH --ntasks=2           # number of tasks
#SBATCH --mem-per-cpu=4GB  # Memory per CPU

module load myModule
srun mpi_binary
```

Also see `https://wiki.auckland.ac.nz/display/CER/Slurm+User+Guide`

# Submitting a Job with Slurm: Send the job to the queue

## Slurm

- To submit a job:

```
sbatch myJob.sl
```

- To monitor your jobs:

```
squeue -u <myUserId>
```

```
squ
```

- To cancel:

```
scancel <jobId>
```

# Notes for Windows Users

- Be careful of Windows end of line (EOL) characters, as Linux applications often handle them poorly.
- MobaXterm has a build in text file editor.
- Notepad++ lets you convert between Windows and Unix style line endings.
- The command line program dos2unix, on the cluster, does the same.
- Even though you can avoid using the Linux command line, having a basic understanding will help you debug your jobs.

# Software

- We have many specialized software packages.
- The best way to see what we have is by checking the wiki.
- The Wiki also has a software section.
- We can install software that you need:
  - Linux version of the software.
  - Command line mode without user interaction.
  - Interaction possible for small tests on the build nodes.
  - We don't provide licenses.
  - You may also install software in your home or project directory.

# How to ask a question

- Mention your UPI and project code.
- What command(s) exactly did you execute?
- What is the error that is printed to the screen?
- If your job failed:
  - What was the job-id?
  - Which SLURM file did you use? (full path)
  - Does it work on the build node?

# Best practices and advice

- Share with us a short test and we will study the scalability of your application.
- Try to be accurate with the wall-time, it will help the scheduler to efficiently schedule the jobs.
- Be aware that you are sharing resources with other researchers.
- A wrong memory request or a wrong job description setup can potentially affect others.
- If your job uses excessive resources or misbehaves, we may be forced to cancel it and inform you by email.

# Our Expectations

## Our Expectations

- We have an acceptable use policy that follows the NeSI IT policies
- We conduct regular reviews of projects to:
  - see how you are going and if you could use some help
  - collect any research outputs from your work on our facility
  - determine how the cluster has helped your research
  - look at the potential for feature stories on your work
- Please contact us if you have any questions
- Please acknowledge us in your publications

# Questions & Answers