Parallel Computing
Exercise session 5

Emil Løvbak & Sahar Chehrazad
emil.loevbak@kuleuven.be & sahar.chehrazad@kuleuven.be

KU Leuven

23 November 2020

Outline

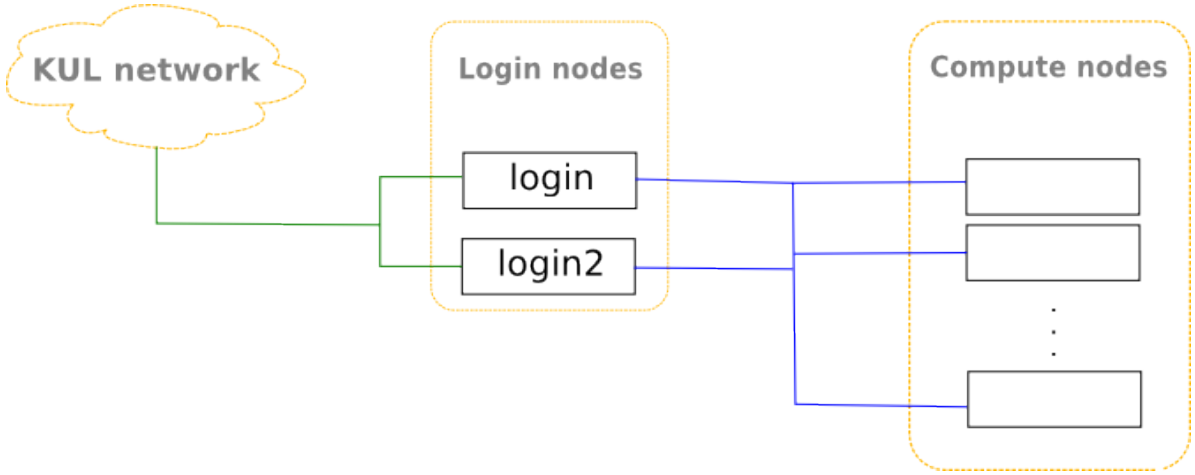
VSC

- VSC is a collaboration between U Antwerpen, VUB, U Gent U Hasselt and KU Leuven
- Rotating investment in new hardware at different institutes
- All institutes can access all hardware
- Two kinds of cluster:
  - Tier-1: Latest, fastest hardware; access only after project is approved
  - Tier-2: Not as new (but still a lot of compute power); Anyone can access if they have/purchase credits
- When a new Tier-1 system is built, the old one becomes a Tier-2 system
- A full overview can be found here.

ThinKing

- Older tier-2 system at KU Leuven/U Hasselt
- Newer system is called Genius
- ThinKing consists of:
  - 160 ivybridge nodes
    - 2 Xeon E5-2680 v2 CPUs@2.8 GHz, 10 cores each
    - 64 GB RAM (130 nodes) / 128 GB RAM (30 nodes)
  - 144 haswell nodes
    - 2 Xeon E5-2680 v3 CPUs@2.5 GHz, 12 cores each
    - 64 GB RAM (32 nodes) / 128 GB RAM (96 nodes)
  - 5 GPGPU nodes
    - 2 Xeon E5-2650 v3 CPUs@2.3 GHz, 10 cores each
    - 64 GB RAM
    - 2 NVIDIA K40c@750 MHz, 12 GB GDDR

Getting access



- VSC clusters consist of login nodes and compute nodes
- You can log in to both using ssh, but in general you will work on a login node
- Access to login node:
  `ssh vsc01234@login-thinking.hpc.kuleuven.be`
- Windows devices can use a third party tool such as Putty

Overhead on the cluster

- To run a job on the cluster, you need to be allocated a node
  $\Rightarrow$ Queuing system takes care of fair distribution of resources
- The queuing system needs to be able to schedule jobs
  $\Rightarrow$ Provide a `.pbs` file with resource estimates for your job
- The cluster contains many conflicting software packages
  $\Rightarrow$ Load the ones you want with a module system
- Jobs are scheduled and run in a batch fashion
  $\Rightarrow$ Output is available in files once the job completes

Useful commands

- Copying files to cluster: `scp local_file vsc01234@login-thinking.hpc.kuleuven.be:remote_folder`
- Copying directory from cluster: `scp -r vsc01234@login-thinking.hpc.kuleuven.be:remote_folder local_folder`
- Loading modules: `module load {intel, GCC, ...}`
- Submitting a job: `qsub -A lp_edu_alg_parallel_comp_2021 job_name.pbs`
- Checking your jobs: `qstat`
- Estimate a job's start time: `showstart jobID`
- Check directory contents: `ls`
- Copy file/directory: `cp`
- Change directory: `cd`
- Find out more about a command: `man command`
- Edit file: `nano file_name`
- Other useful commands: `pwd, mv, mkdir, touch, rm,...`

An example PBS file

- This file defines a job on 4 nodes for 1 minute, requests 10 processors per node and loads the most recent Intel toolkit:

```
#!/bin/bash -l
#PBS -l nodes=4:ppn=10
#PBS -l walltime=00:01:00
module load intel

cd $PBS_O_WORKDIR

./myprog
```

OpenMP

1. Download `omp_hello.c` and `hello_omp.pbs`
2. Copy these files to the cluster using `scp`
3. Compile the program using
   `g++ -o hello omp_hello.c -fopenmp`
4. Check that you understand the file `hello_omp.pbs`
5. Submit the job using `qsub -A lp_edu_alg_parallel_comp_2021 hello_omp.pbs`
6. Check your results

MPI

1. Download `mpi_hello.c` and `hello_mpi.pbs`
2. Copy these files to the cluster using `scp`
3. Load the Intel MPI toolkit using `module load intel`
4. Compile the program using
   `mpiicpc -o hello mpi_hello.c`
5. Check that you understand the file `hello_mpi.pbs`
6. Submit the job using `qsub -A`
   `lp_edu_alg_parallel_comp_2021 hello_mpi.pbs`
7. Check your results