

Introduction to HPC and Linux

For deep learning practitioner

Topics to be covered

- Access Massive
- Basic Linux
- Slurm
- software module
- Job usage monitoring
- Visualization
- Transfer data
- Typical deep learning project workflow

Connect to cluster

Login

- You need to setup your password at karaage
- You will be asked for password for the first time

Secure shell (ssh)

replace this with your username, you can find it at https://hpc.erc.monash.edu.au/karaage/profile/personal/



ssh luhanc@m3.massive.org.au



The hostname/IP address, it can be either:

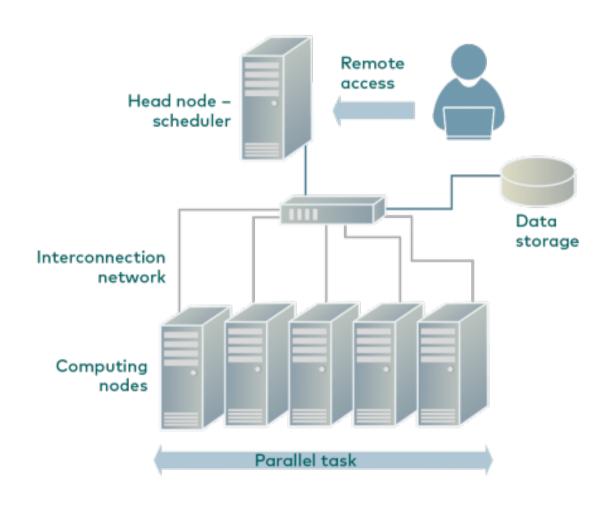
- m3.massive.org.au
- m3-login1.massive.org.au
- m3-login2.massive.org.au
- m3-dtn.massive.org.au

Setup quick access (optional)

- While you are waiting, please have a look at the setup https://
 docs.github.com/en/github/authenticating-to-github/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent
- Successful setup should enable you to login by simply type 'ssh massive'

Overview of cluster

High-performance Cluster



Basic Linux commands

Please open the Linux cheat sheet and follow the activity Feel free to shout out any question

The following command copy training resource to your home directory

/projects/vf38/introHPC/

Data transfer

```
scp <files> <username>@<hostname>:<target location>
```

```
scp *.py luhanc@m3.massive.org.au:/home/luhanc/workshop/
```

Script

- Scripting is basically executing a list of commands line by line
- For example

```
#!/bin/bash
echo current date is `date`
echo my hostname is `hostname`
```

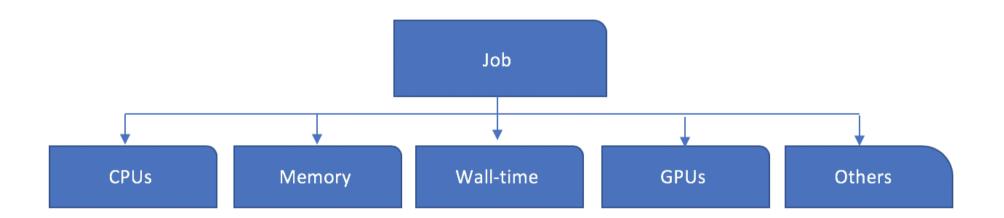
Work remotely (optional)

sshfs -o allow_other,defer_permissions,follow_symlinks <username>@<hostname>:<remote directory> <local directory>

sshfs -o
allow_other,defer_permissions,follow_symlinks,auto_cache,reconnect,defer_permissions,noappledo
uble,nolocalcaches,no_readahead luhanc@m3:/home/luhanc /Users/ChenqLuhan/mount-remote/m3/home

Job scheduling

Component of Job



Example slurm script

Options available at https://slurm.schedmd.com/sbatch.html

```
#!/bin/bash

#SBATCH --time=00:00:10
#SBATCH --mem=1MB
#SBATCH --cpus-per-task=1
#SBATCH --ntasks=1
echo 'echoin' ...'
echo 'current date is '
date
echo 'my hostname is '
hostname
```

Resource is specified by #SBATCH header

Tips

- Running `show_cluster` print the status of machine
- You can check your job through `show_job`
- If your job takes forever to schedule, check if you misspecified some config (e.g. time is 1 day by default)
- specify mail-user/mail-type can often be quite useful

Software Modules

Module Commands

Command	Description
module avail	Show available modules
module list	List loaded modules
module load <i>modulename</i>	Load a module into the current environment
module rm unload <i>modulename</i>	Unload a module from the environment
module purge	Unload all loaded modules
module swap module1 module2	Swap a loaded module with another
module show modulename	Give help for a particular module
module help	Show module specific help

Software Installation

- Existing packages may not meet your requirement
 - Option 1: Email <u>help@massive.org.au</u>
 - Option 2: Install required software by yourself
 - Issue: you cannot run `sudo` on cluster
 - solution: python virtual environment

Python virtual environment

- Two dominate approach
 - conda: You gain lots of speed advantage, but generally harder to manage
 - pip: Simple, widely supported

Python virtual environment

- Create venv
 - python3 -m venv <path to env>
- Enter environment
 - source activate <path to env>/bin/activate
- Install packages
 - pip install <package name>
- Exit environment
 - deactivate

Monitor Job Usage

htopFor CPU and memory

```
- luhanc@m3-login1:~ — ssh m3
                                                      13 [l
14 [l
15 [l
                     0.0%
                                                0.0%
                                                                                 19 [
                                                                                                      1.9%
                     3.9%
                            8
                                                                                 20 [1
                                                0.0%
                                                                          6.5%
                                                                                                      0.7%
                     9.8%
                            9
                                                0.0%
                                                                          1.3%
                                                                                 21
                                                                                                      0.0%
                     5.9%
                                                      16
                                                                                 22
                            10
                                                0.0%
                                                                          0.0%
                                                                                                      0.0%
                     0.6%
                            11
                                                0.6%
                                                      17
                                                                          0.7%
                                                                                 23
                                                                                                      0.0%
                     0.0%
                            12
                                                0.0%
                                                      18 [
                                                                           0.0%
                                                                                 24
                                                                                                      0.0%
 16.1G/108G
                                                      Tasks: 1109, 229 thr; 1 running
                                                      Load average: 0.29 0.28 0.31
                                               0K/0K
                                                       Uptime: 11 days, 13:59:33
                            1596 S 0.0 0.5
                                          1:05.32 /usr/lib/systemd/systemd --switched-root --system --deserialize 22
32563 mhagan
             20
                 0
                  110M
                        176
                               0 S 0.0 0.0 0:00.00 → bash -s --
             20
                        72
                               0 S 0.0 0.0 0:00.00 | L sleep infinity
32596 mhagan
                0
32540 amehnert
             20
                 0
                   110M
                        176
                              0 S 0.0 0.0 0:00.00 ⊢ bash -s --
32574 amehnert
             20
                 0
                   105M
                         76
                               0 S 0.0 0.0 0:00.00 | - sleep infinity
32450 robe0002
             20
                   110M
                        180
                               0 S 0.0 0.0 0:00.01 |- bash -s --
                 0
32495 robe0002
             20
                0
                   105M
                         76
                              0 S 0.0 0.0 0:00.00 | - sleep infinity
32438 s00270745
             20
                0
                   110M
                        1212
                            1036 S 0.0 0.0 0:00.00 → bash -s --
32487 s00270745
             20
                        352
                0
                             280 S 0.0 0.0 0:00.00 | - sleep infinity
             20
                        172
32344 jwriggle
                   110M
                0
                              20
                   105M
                         76
                               32379 jwriggle
                 0
32340 robe0002
             20
                        176
                   110M
                              0
32413 robe0002
             20
                   105M
                         76
                0
                               1040 S 0.0 0.0 0:00.00 ⊢ bash -s --
32290 ksabaroe
             20
                0
                   110M
                        1212
32323 ksabaroe
             20
                0
                   105M
                        352
                             280 S 0.0 0.0 0:00.00 | - sleep infinity
32196 rjoh0016
             20
                0
                   110M
                        172
                              0 S 0.0 0.0 0:00.01 |-- bash -s --
32229 rjoh0016
             20
                0
                         72
                              20
32075 ymuniand
                0
                   110M
                        1212
                            20
32110 ymuniand
                0
                        356
                             280 S 0.0 0.0 0:00.00 | - sleep infinity
32052 s00270745
             20
                        1212
                0
                   110M
                            1036 S 0.0 0.0 0:00.00 ├ bash -s --
32087 s00270745
            20
                0
                        356
                             280 S 0.0 0.0 0:00.00 | - sleep infinity
31953 mhos0007
             20
                0
                  110M
                        1212
                            1036 S 0.0 0.0 0:00.00 ⊢ bash -s --
31986 mhos0007
             20
                        352
                             280 S 0.0 0.0 0:00.00 | L sleep infinity
                0
                        172
31840 clau
             20
                0 110M
                              31876 clau
             20
                0 105M
                               31775 jbak
             20
                0 110M
                        172
                               0 S 0.0 0.0 0:00.01 ├ bash -s --
31808 jbak
             20
                0 105M
                        76
                               31658 earsenau 20
                0 110M
                        172
                              0 S 0.0 0.0 0:00.00 ├ bash -s --
F1Help F2Setup F3SearchF4FilterF5SortedF6CollapF7Nice -F8Nice +F9Kill F10Quit
```

nvidia-smi

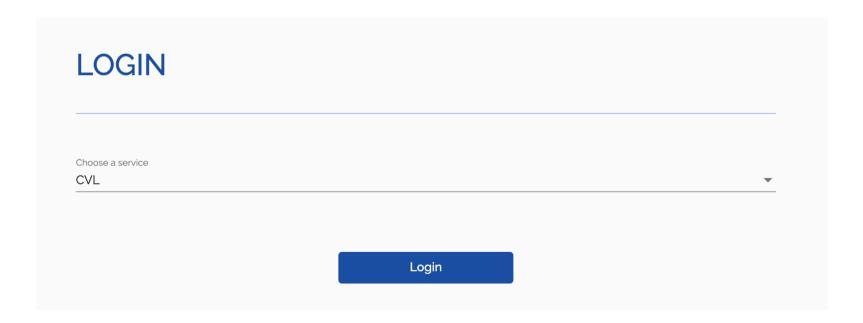
Maximizing GPU utility by pushing this value to its max

```
[luhanc@m3h003 ~]$ nvidia-smi
Wed Jul 8 03:29:40 2020
 NVIDIA-SMI 418.40.04
                         Driver Version: 418.40.04
                                                       CUDA Version: 10.1
                  Persistence-MI Bus-Id
                                                Disp.A | Volatile Uncorr. ECC
 GPU Name
                                                         GPU-Util Compute M.
            Perf Pwr:Usage/Capl
                                          Memory-Usage
      Temp
   0 Tesla P100-PCIE...
                                 00000000:00:08.0 Off
       30C
                     25W / 250W
                                       0MiB / 16280MiB
                                                              0%
                                                                      Default
                                                                   GPU Memory
 Processes:
  GPU
            PID
                                                                   Usage
                   Type
                          Process name
  No running processes found
```

Visualization and Strudel2

Remote Desktop for Graphics

- Go to https://beta.desktop.cvl.org.au/login
- Select CVL as the service then login using AFF with your monash account



conda environment

Creating your conda environment

module load conda-install
conda-install <absolute path of installation directory>
source <absolute path of installation directory>/bin/activate

Install necessary packages for desktop

conda install -c conda-forge -y jupyter jupyterlab

A MNIST Example

Deep Learning + GPU + Multi-node

 Existing tools generally take care of multi-processing (OpenMP) and GPU (CUDA) well. But Multi-node (MPI) computing is more convoluted.

Copy data is expensive, optimisation means reduce $\frac{communication}{computation}$

Data parallel vs model parallel

Question