Parallel Machine Learning and Artificial Intelligence

Prof. Handan Liu, PhD

h.liu@northeastern.edu

Northeastern University

Spring 2020

Content

- Introduction to the Discovery cluster
- Using Discovery
  - Connecting to Discovery
  - Data transfer
  - Loading Module
  - Using Slurm
  - Running jobs: interactive mode and batch mode; Job scripts
- Linux Fundamentals for Discovery Cluster
- Learn how to compile and run OpenMP and MPI programs (in C)

Connecting to Discovery

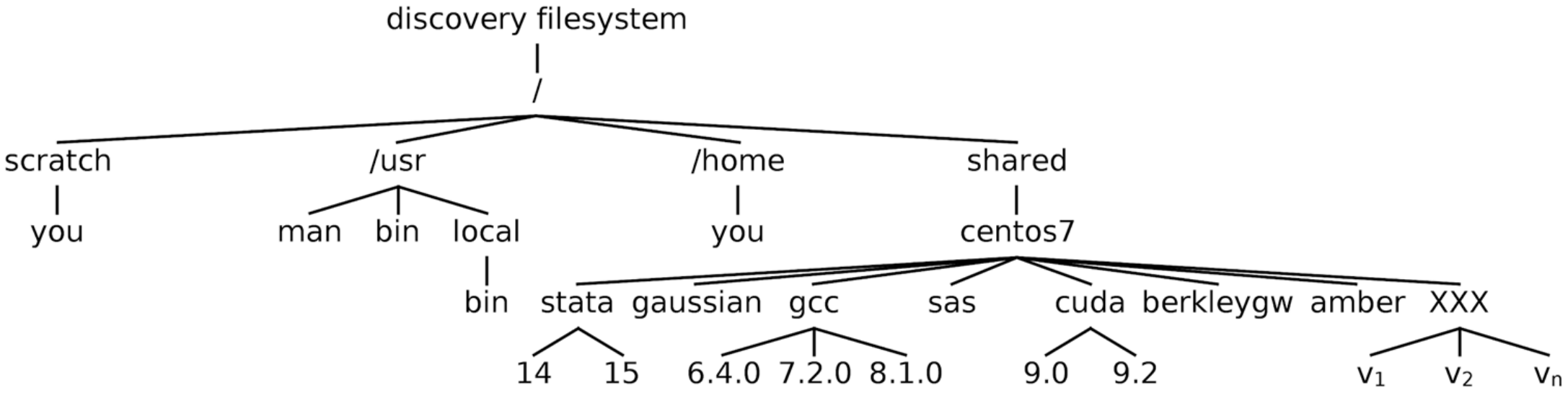
- Secure SHell (SSH)
  - Linux/Mac: Terminal
  - Windows: Putty, MobaXTerm

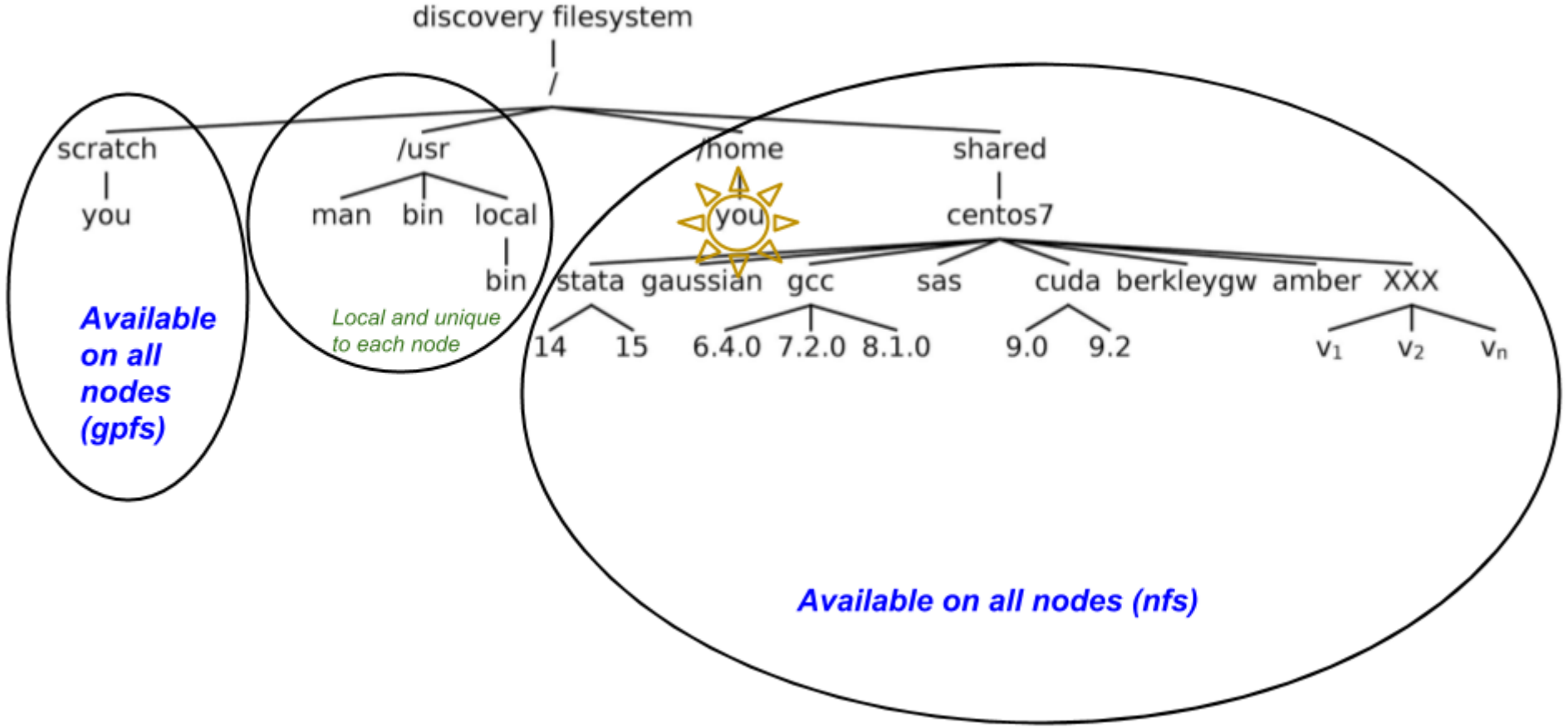
$ ssh <username>@login.discovery.neu.edu

  - Here, username is your Northeastern username

- Graphical User Interface (GUI), optional
  - Mac: Xquartz
  - Windows: X11

Discovery Filesystem




discovery filesystem
/
scratch
you
/usr
man
bin
local
bin
/home
you
shared
centos7
stata
gaussian
gcc
sas
cuda
berkleygw
amber
XXX
14
15
6.4.0
7.2.0
8.1.0
9.0
9.2
v1
v2
vn
Available on all nodes (gpfs)
Local and unique to each node
Available on all nodes (nfs)

Data Transfer

- On command line (Linux like terminal) – scp, rsync
- Graphically – FileZilla
- Run the commands on your local machine:

  $ scp (-r for folder) source destination

  $ rsync -auv local remote
- Remote address:

  username@xfer.discovery.neu.edu

  sftp://xfer.discovery.neu.edu
- See RC document.

Documentation

- Almost all utilities have the --help option which shows what the options are
- man utility shows the manual page for the utility, or a system call or any concept.
  - q → quit

Filesystem navigation: viewing

- `ls` - list files
  - Options: ls -[lart1d] most common options
    - -`l`: long listing
    - -`a`: all files (including hidden files, (ones that begin with a .))
    - -`r`: show in reverse order (default sorting order is alphanumeric)
    - -`t`: sort by time modified
    - -`1`: show all in 1 column
    - -`d`: show directory names but not their content

Viewing file content

`cat`: print content of file to the stdout (more on streams later)

`more`: print content of file to the screen but pause between pages

`less`: like more but with more options

`vi/vim/gedit`: open file in an editor

File and directory manipulation

copy : `cp [-rpiv]`

- files: `cp file1 file2`
- Directories: `cp -r dir1 dir2`

move/rename: `mv oldpath/oldname newpath/newname`

Remove: `rm -[irf] / rmdir -[pmv]`

- Files: `rm file1 file2 file3 …. filek filen`
- Directories: `rmdir dir1 dir2 dir3 … dirk dirn`

Find a file in your directory tree

I want to find all the files which end with .err:

```
find . -type f -name '*.err'
```

- Using `find --help` or man find to get more information

Archives

Sometimes it may be necessary to pack up an entire directory into a single file, for archiving, reducing the number of files in the filesystem, or easy transfer.

Create an archive from directory foo:

```
tar cvf foo.tar ./foo
```

```
c: create, f: name of file to create
```

Unpack an archive:

```
tar xvf foo.tar
```

```
x: xtract, v: verbose, f: filename of the archive
```

Archives

Often tar archives are compressed to form what is known in the uinx/linux world as a “tarball”

To create or unpack a tarball, use the z option

```
tar czvf foo.tgz ./foo

tar zxvf foo.tgz

zip creates archives compatible with windows archives
```

Slurm

- Slurm (Simple Linux Utility Resource Management) is the scheduling software that you use to schedule your jobs on Discovery.
- See the RC document.
- The sinfo shows the basic information of the cluster:

```
$ sinfo
$ sinfo –p partition-name
```

- Then scontrol shows configuration of partitions and nodes.

```
$ scontrol show res=reservation-name
$ scontrol show partition=partition-name
$ scontrol show nodes=node-name
```

Using Module

Running jobs

- Single node;
- Multiple nodes.
- Interactive mode
- Batch mode
- Move to a compute node by using `srun` or `sbatch`
- You should **NEVER** launch any jobs from the login nodes login-00 or login-01. Any job launched from the login node will be terminated.

Running jobs – Interactive

- Allocate resource, log onto compute node, run the job and exit
- To move to a compute node, at the command prompt type:

```
$ srun --pty /bin/bash
$ srun –p partition-name --pty /bin/bash
$ srun --reservation=CSYE7374 --pty
/bin/bash
```

- Easy /small / debug
- Interactive

Running jobs – Batch

- Submit to the cluster for later execution

  `$ sbatch` configuration file

- Configuration file
  - Parameters, what resource do you want
  - Commands, instructions to be executed

- A little more effort
- For longer and bigger jobs

Submitting a job

- See the RC document.
- The general format for submitting a job to the scheduler is as follows:

```
$ sbatch example.sbatch
```

- Example Job Scripts

Sample batch script

```
#!/bin/bash

## Both in single-letter and whole-word formats, e.g. -N 1 and --nodes=1
## Normal configurations
#SBATCH --job-name=test
#SBATCH --output=test.out
#SBATCH --error=test.err
#SBATCH --time=00:15:00

## Parallel configurations
#SBATCH -p general
#SBATCH -N 1
##SBATCH -n 2   #OR --ntasks-per-node=10
##SBATCH --cpus-per-task=2
##SBATCH --mem-per-cpu=50G
```

Constraint options

#SBATCH --mem=10G

#SBATCH --nodelist=c1[234-238]

#SBATCH --exclude=c1234

#SBATCH --constraint="E5-2690v3@2.60GHz"

Dependencies options

#SBATCH --dependency=after:jobid

#SBATCH --dependency=afterok:jobid

#SBATCH --dependency=afternotok:jobid

Load modules

Change directory

Start the job

Monitoring - Jobs

- squeue, jobs in the queue

```
$ squeue -u $USER -p partition -t PENDING
```

- scontrol, detailed job information,

  only for active jobs

```
$ scontrol show job id
```

- seff, efficiency related statistics,

  only for completed jobs

```
$ seff jobid
```

Parallel Examples

How to compile and run an OpenMP Program

Compiler	Compiler Options	Default behavior for # of threads (OMP_NUM_THREADS not set)
GNU (gcc, g++, gfortran)	-fopenmp	as many threads as available cores
Intel (icc ifort)	-openmp	as many threads as available cores
Portland Group (pgcc,pgCC,pgf77,pgf90)	-mp	one thread

GNU Compiler Example:

$ **gcc -o** omp_helloc **-fopenmp** omp_hello.c
$ export OMP_NUM_THREADS=2
$ ./omp_helloc

Intel Compiler Example:

$ **icc -o** omp_helloc **-openmp** omp_hello.c
$ export OMP_NUM_THREADS=3
$ ./omp_helloc

How to Compile and Run a MPI Program

The table below lists OpenMPI compiler wrapper scripts for Linux clusters.

Language	Script Name	Underlying Compiler
C	**mpicc**	C compiler for loaded compiler package
C++	**mpiCC** **mpic++** **mpicxx**	C++ compiler for loaded compiler package
Fortran	**mpif77**	Fortran77 compiler for loaded compiler package. Points to mpifort.
	mpif90	Fortran90 compiler for loaded compiler package. Points to mpifort.
	mpifort	Fortran 77/90 compiler for loaded compiler package.

Compile and Run your cases

Special tips – Install Python package

- Use the anaconda on cluster

```
$ module load anaconda
$ conda install package
```

- Install it on your own (download the source code)

```
>>> import sys
>>> sys.path.append(path_to_the_package)
```

- Install your own anaconda/virtual env/…

The End!

Assignment:

1. Practice Linux/Slurm commands on Discovery.
2. Don’t run any job on login nodes!!!!