Getting onto MIDWAY

Simon Scheidegger
simon.scheidegger@gmail.com
Thursday, July 20th 2017
Open Source Macroeconomics Laboratory Boot Camp – BFI/UChicago

Outline

- Make first steps on a Linux Cluster
  Login via ssh, remotely, short overview of basic unix commands like cd, pwd, cp, scp,...

- How to submit and manage jobs / computations

https://rcc.uchicago.edu

For this course, we use the Uchicago's **Midway compute cluster**.

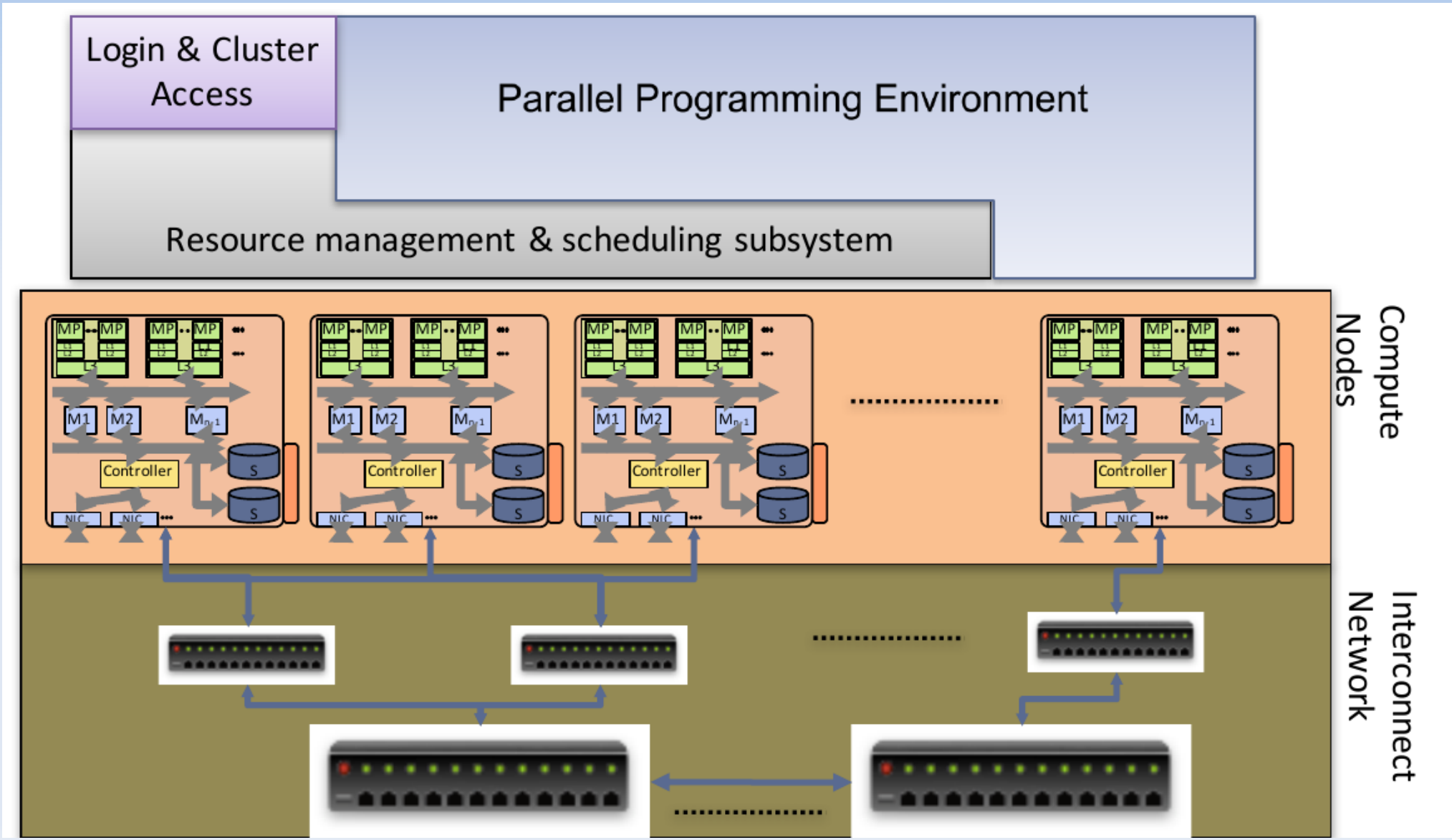
It is managed by the **R**esearch **C**omputing **C**entre

→ Its setup is very similar to any other top system

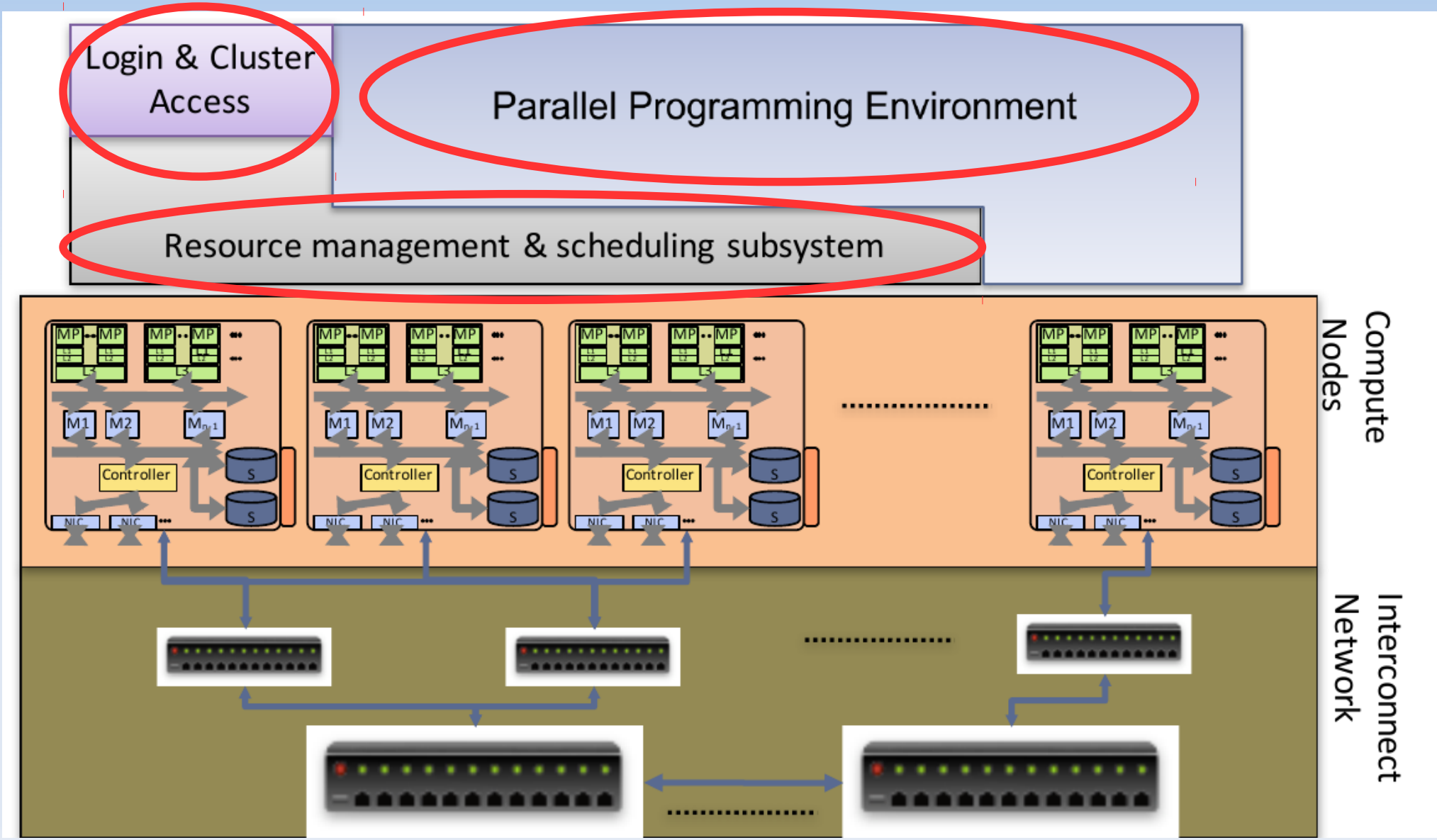
For the RCC Manual see the documentation site at

http://docs.rcc.uchicago.edu

An abstract compute cluster



An abstract compute cluster



The size of a HPC cluster

Credit – https://bluewaters.ncsa.illinois.edu/

Login for participants

- If you don't have an account on Midway request one (infrastructure for this course)
  https://rcc.uchicago.edu/docs/using-midway/index.html

- For MS-Windows users: Download and install Putty
  → http://the.earth.li/~sgtatham/putty/latest/x86/putty.exe

  → Download and install Winscp

  → http://winscp.net/download/winscp576setup.exe

Basic Linux commands (1)

Command	Description
pwd	Print name of current/working directory
cd [Directory]	Change directory (no directory → change to home)
ls [Directory]	List directory contents (no directory → list current)
cat FILE	Concatenate files and print on the standard output
mkdir DIRECTORY	Make directories
mkdir -p DIRECTORY	Make directories, make parent directories as needed
cp SOURCE... DIRECTORY	Copy files and directories
cp -r SOURCE... DIRECTORY	Copy files and directories, copy directories recursively
mv SOURCE... DIRECTORY	Move (rename) files
man COMMAND	An interface to the on-line reference manuals

Basic Linux commands (2)

Command	Description
ssh -X foo@host.com	OpenSSH SSH client (remote login program), access to host.com with user foo
scp foo@host.com:/home/bar ./	Secure copy (remote file copy program), copy file bar from /home on host.com to directory
scp bar foo@host.com:/home/	Secure copy (remote file copy program), copy file bar from the local host to /home on host.com
git clone git@github.com:whatever folder-name	The stupid content tracker, Clone a repository (whatever) into a new directory (folder-name).
git checkout	Checkout a branch or paths to the working tree.

Other clusters – Step-by-Step

- First login, get lecture notes

  (MS-Windows: Putty, Linux/MacOS: Terminal)

```
> ssh -X USERNAME@midway1.rcc.uchicago.edu
> git clone ***lecture-folder*** #clone lecture
> cd ***lecture_folder***  #go into folder
> ls # list content of folder
```

Step-by-Step (2)

→ **Perform some basic operations on the cluster**

```
> ssh -X USERNAME@hpc.alphacruncher.net
> pwd
/home/USERNAME
> mkdir -p firstFolder/secondFolder
> ls
FirstFolder
> ls firstFolder
secondFolder
> cd firstFolder
> pwd
/home/USERNAME/firstFolder
> ls
secondFolder
> exit
```

Step-by-Step (3)

- How to copy folders and files to your PC?
- MS-Windows, start WinSCP

  → Host-Name: `midway1.rcc.uchicago.edu`

  → User: `USERNAME`
- Linux/MacOS, replace /YOUR-LOCAL-PATH/

  → with /home/LOCAL-LOGIN-NAME/ for linux

  → with /Users/LOCAL-LOGIN-NAME/ for MacOS

```
>scp -r USERNAME@midway1.rcc.uchicago.edu:/home/simonsch/YOUR-LOCAL-PATH/ .
```

Step-by-Step (4)

- Copy folders and files from your notebook

  create a file named firstFile in firstFolder

  → MS-Windows: use WinSCP to copy the directory back

  → Linux/MacOS

```
>scp -r /YOUR-LOCAL_PATH/firstFolder/FILENAME USERNAME@midway1.rcc.uchicago.edu:
```

- Check that file is there by

```
>ssh -X midway1.rcc.uchicago.edu
> ls
FILENAME
>cat FILENAME  #shows content of file
```

Environment setup

Supporting diverse user community requires supporting diverse tool sets (different vendors, versions of compilers, debuggers, libraries, apps, etc)

User environments are customized via modules system (or softenv)

```
> module avail #shows list of available modules

> module list  #shows list of modules loaded by user

> module load module_name #load a module e.g. compiler

> module unload module_name #unload a module
```

Using an editor on a cluster

Compute clusters like Midway's infrastructure have a variety of simple text editors available.

→ vi, vim

```
>vi helloworld.cpp

#include <iostream>

int main()
{
  std::cout << "Hello World!"  << std::endl;

  return 0;
}
```

More low bandwidth editors

Depending on network and preference, you may want to use an editor without a graphical user interface; common options:

- vi/vim
- emacs
- nano

emacs:

Undo: C-_

Find/create file: C-x C-f

Save file: C-x C-s

Exit Emacs: C-x C-c

Quit: C-g

Two modes – insertion and command mode
Insertion mode begins upon an insertion
[ESC] returns to command mode

Command mode options:
:w save
:wq save and exit
:q exit as long as there are no changes
:q! exit without saving

Insertion:
i (insert before cursor)
a (append)

Deletion: x
Motion: h (left) k (up)
j (down) l (right)

Slurm Workload Manager

http://slurm.schedmd.com/

Simple Linux Utility for Resource Management (SLURM).

Open-source workload manager designed for Linux clusters of all sizes.

Provides three key functions:

1) It allocates exclusive and/or non-exclusive access to resources (computer nodes) to users for some duration of time so they can perform work.

2) It provides a framework for starting, executing, and monitoring work (typically a parallel job) on a set of allocated nodes.

3) It arbitrates contention for resources by managing a queue of pending work.

```
> sbatch submit_helloworld.sh (submit job)
> squeue -u NAME   (status of job)
> scancel JOBID      (cancel job)
```

Run an executalbe with „slurm“

https://rcc.uchicago.edu/docs/running-jobs/index.html#running-jobs

```
#!/bin/bash -l

#SBATCH --ntasks=1     ## how many cpus used here

#SBATCH --time=01:00:00  ## walltime requested

#SBATCH --output=slurm_test.out ## output file
#SBATCH --error=slurm_test.err     ## error


### executable
./helloworld.exe
```

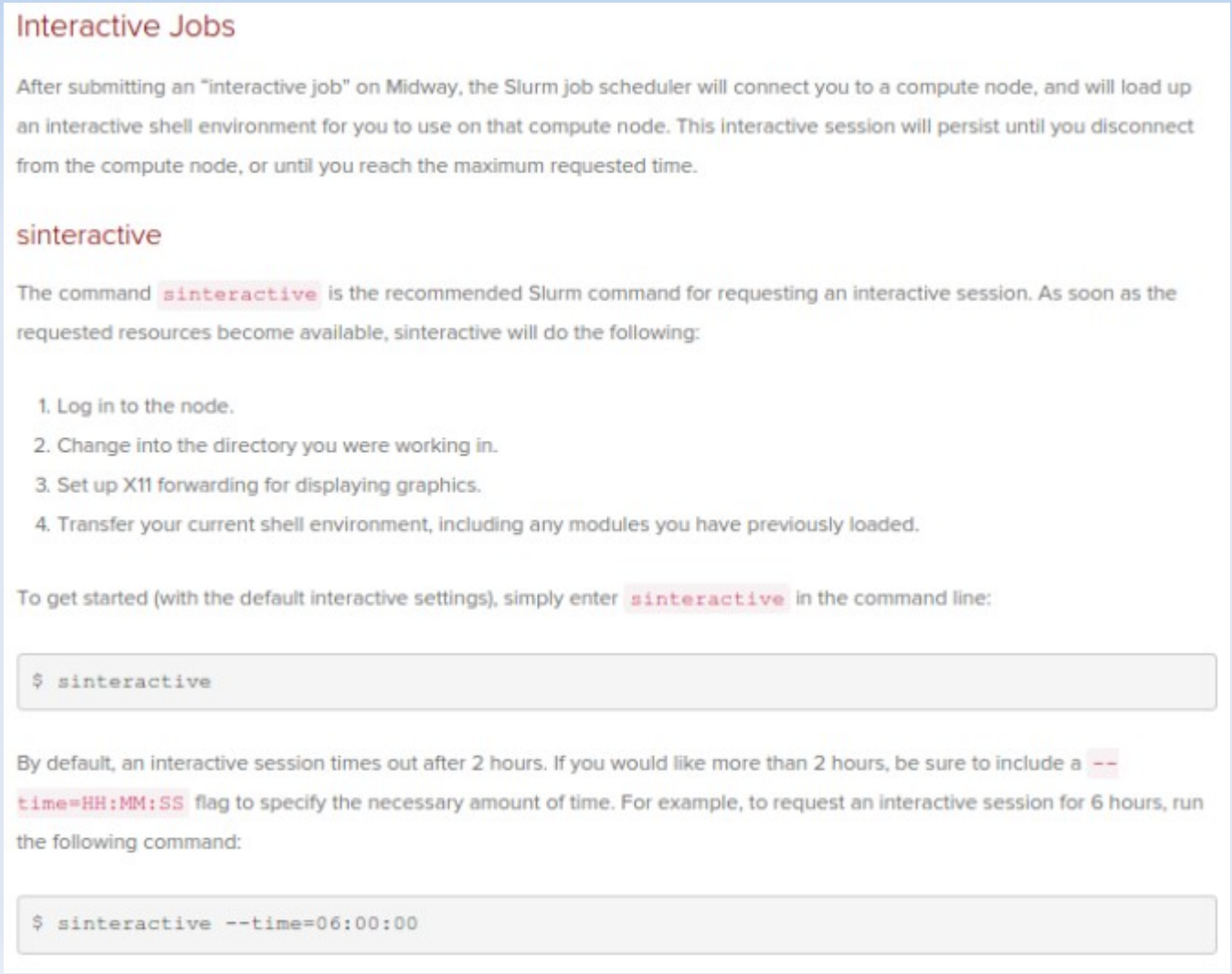
Run an executalbe on MIDWAY

https://rcc.uchicago.edu/docs/running-jobs/index.html#running-jobs

```
#!/bin/bash
# a sample job submission script to submit an MPI job to the sandyb partition on Midway1

# set the job name to hello-world
#SBATCH –job-name=hello-world

# send output to hello-world.out
#SBATCH --output=hello-world.out

# receive an email when job starts, ends, and fails
#SBATCH --mail-type=BEGIN,END,DAIL

# this job requests 1 core. Cores can be selected from various nodes.
#SBATCH --ntasks=1

# there are many partitions on Midway1 and it is important to specify which
# partition you want to run your job on. Not having the following option, the
# sandby partition on Midway1 will be selected as the default partition
#SBATCH --partition=sandyb

# Run the process with mpirun. Notice -n is not required. mpirun will
# automatically figure out how many processes to run from the slurm options
./helloworld.exe
```

Interactive sessions on MIDWAY

- https://rcc.uchicago.edu/docs/using-midway/index.html#interactive-jobs



Interactive Jobs

After submitting an "interactive job" on Midway, the Slurm job scheduler will connect you to a compute node, and will load up an interactive shell environment for you to use on that compute node. This interactive session will persist until you disconnect from the compute node, or until you reach the maximum requested time.

sinteractive

The command `sinteractive` is the recommended Slurm command for requesting an interactive session. As soon as the requested resources become available, sinteractive will do the following:

1. Log in to the node.
2. Change into the directory you were working in.
3. Set up X11 forwarding for displaying graphics.
4. Transfer your current shell environment, including any modules you have previously loaded.

To get started (with the default interactive settings), simply enter `sinteractive` in the command line:

```
$ sinteractive
```

By default, an interactive session times out after 2 hours. If you would like more than 2 hours, be sure to include a `--time=HH:MM:SS` flag to specify the necessary amount of time. For example, to request an interactive session for 6 hours, run the following command:

```
$ sinteractive --time=06:00:00
```