A Hands-On Tutorial:
RNA Sequencing Using High-Performance Computing
February 11th and 12th, 2016

1st session (Thursday)
Preliminaries: Linux, HPC, command line interface
Using HPC: modules, queuing system
Presented by: Ali Siavosh, Eric Peskin, Loren Koenig

2nd session (Friday)
RNA sequencing: basic concepts and data preparation, using HPC to analyze the data
Presented by: Adriana Heguy, Stuart Brown
HPC (High-Performance Computing)? generally refers to the practice of aggregating computing power in a way that delivers much higher performance than what one could get out of a typical desktop computer or workstation in order to solve large problems in science, engineering and business.

Phoenix:
- **Head node:**
  32 cores, 128 GB RAM.
- **70 Compute “worker” nodes:**
  Each Compute node: 32 cores, 128 GB RAM.
login

ssh demo@phoenix.med.nyu.edu

ssh Keys

logout
## Linux Operating System

**Graphical User Interface (GUI) vs. Command Line Interface (CLI):**

### Commands:

- **pwd**
  
  `/ifs/home/siavoa01`

- **ls**
  
  - `ls`
  - `man ls`

**Root directory, denoted by a single slash**

```
/  
  bin/  ifs/  ...
  data/  home/  ...
  tutorial/  ...  user01/  ...
  project1/  
```

**Working directory**

Absolute path: `/ifs/home/user01`

**Commands:**

- `pwd`
- `ls`
- `man ls`

**Examples:**

- `echo -n Hello world!`
- `ls -l`
Making Directory:  
```
mkdir project1
mkdir project2
```

List of contents in a directory:  
```
ls
ls -l
```

Remove a directory:  
```
rmdir project2
```

Change to a directory:  
```
cd project1
cd ..
```

Editor:  
```
nano script1
```
```
This is my first script ^o ^x
```

Remove file/directory:  
```
rm script1
```
```
nano script2
```
cp script1 project1/script2
mv script1 project1/.
cd project1
cp script1 script2
echo “Hello World!”
date
nano script2
#!/bin/bash
echo “Hello World”
date

cat script2
chmod +x script2
./script2
File transfer

Command line (CLI):
scp myfile userid@phoenix.med.nyu.edu:
scp userid@phoenix.med.nyu.edu:project1/script2.

Graphical User Interface (GUI):
WinSCP
Environment Modules

• Some programs need extensive setup of environment variables.

• We have multiple versions of some programs installed.

• Solution: *Environment Modules* make sure that environment variables are set up for:
  - Software – and version – you want to use.
  - Any prerequisite software.
  - No conflicting software.
Environment Modules: Commands

- module avail  List available modules.
- module list   List currently loaded modules.
- module load *name*  Load named module.
- module unload *name*  Unload named module.
- module help *name*  See help on named module.
- module show *name*  See what module will do to environment.
- man module  Get help on module command itself.
Environment Modules: Versions and Conflicts

```bash
bedtools --version
module avail bedtools
module load bedtools/2.15.0
bedtools --version
module load bedtools/2.17.0
module unload bedtools/2.15.0
module load bedtools/2.17.0
bedtools --version
echo $BEDTOOLS_ROOT
ls $BEDTOOLS_ROOT
```
Environment Modules: Prerequisites

module list
module load macs/1.4.2
module list
module unload macs
module list
module load macs/2.0.10
module list
module unload macs
module list
SGE: The Batch System

HPC Linux Cluster

- Cluster has many compute nodes and many users.
- Need to spread out work among the compute nodes.
- Rather than having users hunt for node that is not too busy...
- ... let the batch system do it!
- We use a batch system called Sun Grid Engine (SGE).
SGE: Cluster Usage

- Users interact with head node of the cluster.
  - Editing, compiling, file manipulation, etc.
- From the head node, users submit **jobs** to batch system.
  - Submission includes **shell script** with commands to run.
- Batch system runs jobs on compute nodes.
- Users collect output in files viewable on the head node.
Example Script

- Use nano to create a file with the following contents, and save it as `simple.bash`

```bash
#!/bin/bash
date
sleep 20
date
```
Running Script

- Make script executable.
  
  `chmod +x simple.bash`

- Run script
  
  `./simple.bash`

- Normally we would not run directly on the head node.

- Don't try this at home!
SGE: Submit Job

- Submit job
  
  ```bash
gsub -S /bin/bash -cwd simple.bash
  ```

- Note job name and job number.

- Monitor status (try the following a few times quickly):
  
  ```bash
  qstat
  ```
  - Watch progression from `qw` (waiting) to `r` (running) to not listed (finished).
SGE: Directives

• Could put SGE directives in script itself

```bash
#!/bin/bash
#$ -S /bin/bash
#$ -cwd
date
sleep 20
date
```
SGE: Output and Error Files

- When job is finished, look for output files.
  
  `ls -ltr`

- Standard output redirected to `jobname.o jobnumber`

- Standard error redirected to `jobname.e jobnumber`

- Check size of .e file.

- Read output (substitute your job number below):
  
  `cat simple.bash.o jobnumber`
SGE: Useful Directives

- `cwd` Run script in Current Working Directory (where you submit).

- `S /bin/bash` Use this shell to interpret script.

- `M first.last@nyumc.org` Send status updates to this address.

- `m when` (when = any combination of following, e.g. `ae`)
  
  `a = abort` Mail if disaster strikes.
  
  `b = begin` Mail when job starts.
  
  `e = end` Mail when job completes.
SGE: Useful Commands

- `qsub` Submit job.
- `qstat` Check status of running jobs or queues.
- `qacct` Get accounting information on finished jobs.
- `qdel` Delete (kill) running job.
- `qlogin` Start interactive shell on compute node.
  - Use sparingly and remember to logout when finished.
Environment Modules in SGE Scripts

• Use needed modules in your SGE scripts

```bash
#!/bin/bash
#$ -S /bin/bash
#$ -cwd
module load samtools/0.1.19
module load bwa/0.7.3a
samtools...
bwa...
```
Multi-threaded Jobs

• Several applications can use multiple *threads*:  
  – a form of shared-memory parallelism  
  – uses multiple CPU cores *within one node*

• Use SGE *threaded parallel environment*. 
Threaded Environment Examples

- `qsub -pe threaded 4 script.sh`
  - waits until 4 cores available on one node
  - starts on that node, claiming 4 slots

- `qsub -pe threaded 6-13 script.sh`
  - waits until at least 6 cores on one node
  - starts on node with most available (up to 13)
  - claims as many as available on that node (up to 13)
How Many Did I Get?

- SGE defines **NSLOTS** variable to tell you:
  ```bash
  #!/bin/bash
  #$ -S /bin/bash
  #$ -cwd
  module load cufflinks/2.1.1
  cufflinks -p $NSLOTS ...
  ```
Threads: Best Practices

• Read application documentation carefully:
  – Are multiple threads supported?
  – Do they happen by default?
  – What are the tradeoffs?
  – What flag controls them?
• Don’t hard-code number of threads.
  – use $NSLOTS
Variables

• What was that $BEDTOOLS_ROOT (or $NSLOTS) anyway?
• A *variable*: A name associated with value
• Can be defined by:
  – System
  – Environment modules
  – SGE
  – You!
Variables: User defined

- *Environment variable* is a name associated with a value.
- To access value, put dollar sign in front of name.
- Try:
  
  ```
  NAME=Bob
  echo $NAME
  NAME=Alice
  echo $NAME
  ```
Variables in script

#!/bin/bash
#$ -S /bin/bash
#$ -cwd
dir=/some/really/long/path
mkdir $dir
mkdir $dir/file.txt $dir/
Arguments in script

#!/bin/bash
#$ -S /bin/bash
#$ -cwd
dir=$1
mkdir $dir
mkdir $dir

qsub script.sh /some/really/long/path
Multiple Arguments in script

#!/bin/bash
#$ -S /bin/bash
#$ -cwd
dir=$1
file=$2
mkdir $dir
cp $file $dir/

qsub script.sh /some/really/long/path file1.txt
Exercises

• Write and submit the following SGE scripts:
  – date/sleep/date script, but with sleep time as argument
  – script that prints value of $NSLOTS
    • Submit it with threaded parallel environment
    • Use range of slots min-max
    • Watch it with qstat: hint: add a sleep command, so you can see it in qstat for a while
    • Check output file
Further Reading and Help

- User guide: https://genome.med.nyu.edu/hpcf/wiki
- Our past (longer) tutorials at http://www.med.nyu.edu/chibi/education/tutorials
  - Linux for Cluster Computing: Two tutorials
- Cornell tutorial: https://cvw.cac.cornell.edu/Linux/
- Kernighan and Pike. *The Unix Programming Environment*
- Ask us: hpc_admins@nyumc.org
- For problems, please include command you typed and the messages that resulted (copy paste session), full path to scripts and output files.