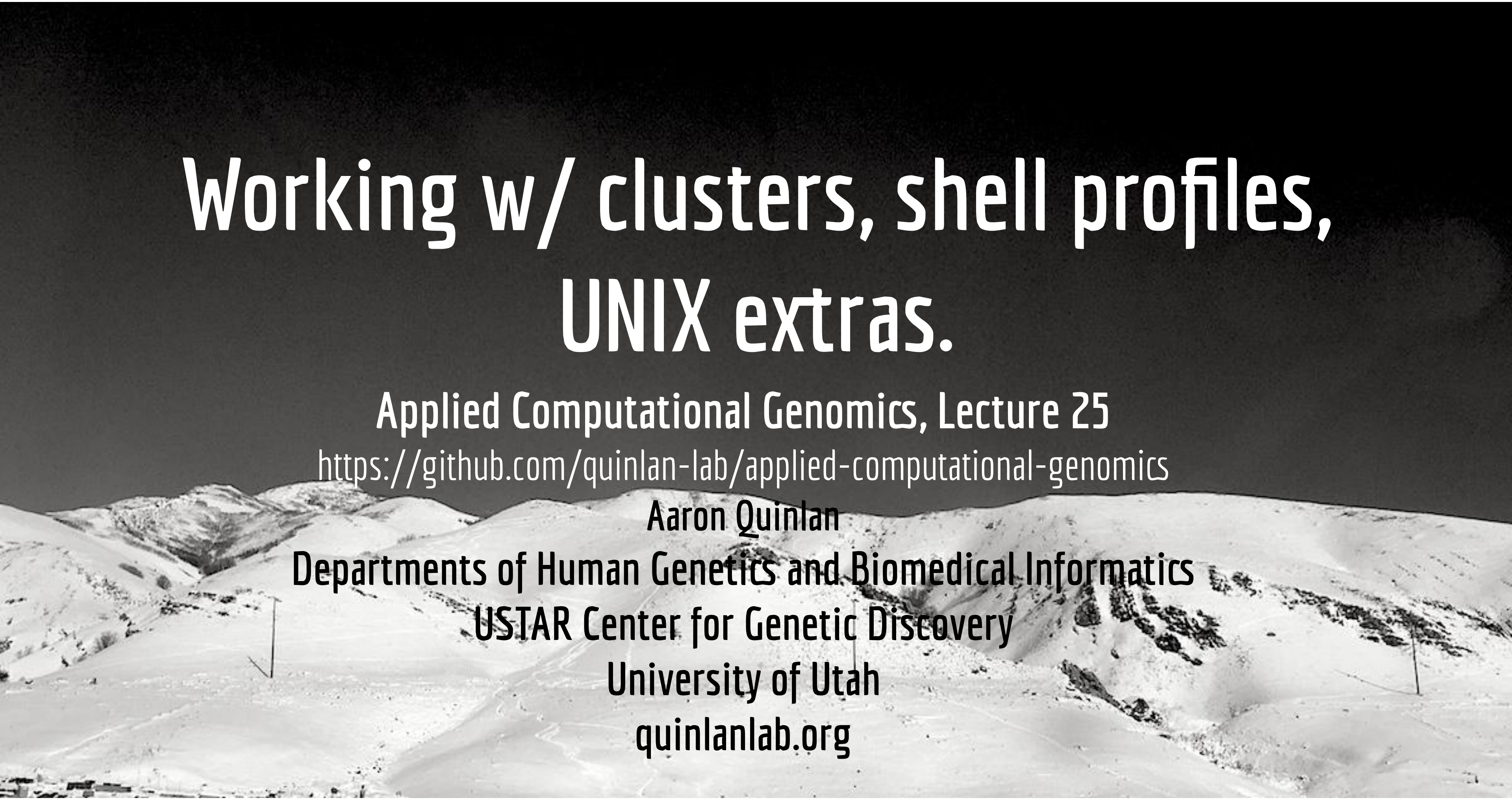


Working w/ clusters, shell profiles, UNIX extras.

Applied Computational Genomics, Lecture 25

https://github.com/quinlan-lab/applied-computational-genomics

Aaron Quinlan
Departments of Human Genetics and Biomedical Informatics
USTAR Center for Genetic Discovery
University of Utah
quinlanlab.org

Revisiting Unix tools and maybe some new ones

- `ps`
- `top`
- `kill`
- `diff`
- `sleep`
- `chmod`
- `history`
- `Ctrl+R`

.bash_profile : run a set of commands each time you login

```
cat ~/.bash_profile

echo "Hi Aaron. What’s the criac?"
# Get the aliases and functions
if [ -f ~/.bashrc ]; then
    . ~/.bashrc
fi
```

.bash_profile versus .bashrc

- .bash_profile is executed each time you login to a machine with a username and password.
- .bashrc is executed each time you open a new terminal window once already logged in.
- The exception is OSX - it always calls .bash_profile

```
cat ~/.bashrc

PATH=$PATH:~u6000771/bin

alias ll='ls -ltr'
alias grep='grep --color'
```

The UNIX `ps` command

```
NAME
       ps - report process status

SYNOPSIS
       ps [options]

DESCRIPTION
       ps  gives a snapshot of the current processes. If you want
       a repetitive update of this status, use top. This man page
       documents the /proc-based version of ps, or tries to.
```

```
# get details about the processes I have running on this machine
ps -ef | grep u1007787
```

The UNIX `ps` command

```
sleep 100

# get details about the processes I have running on this machine
ps -ef | grep u1007787
root     12767  4286  0 06:05 ?        00:00:00 sshd: u1007787 [priv]
u1007787 13106 12767  0 06:05 ?        00:00:00 sshd: u1007787@pts/0
u1007787 13107 13106  0 06:05 pts/0    00:00:00 -bash
u1007787 48272 13107  0 06:46 pts/0    00:00:00 sleep 100
u1007787 48791 13107 11 06:46 pts/0    00:00:00 ps -ef
u1007787 48792 13107  0 06:46 pts/0    00:00:00 grep --color u1007787
```

The UNIX `kill` command

```
# Use kill command to terminate a process. First get the process
id using ps -ef command, then use kill -9 to kill the running
Linux process as shown below. You can also use killall, pkill,
xkill to terminate a unix process.

$ ps -ef | grep bedtools
arq5x    8945  7222  9 22:43 pts/2    00:00:00 bedtools

$ kill -9 8945
```

The UNIX top command

```
NAME
       top - display Linux processes



SYNOPSIS
       top -hv|-bcHiOSs -d secs -n max -u|U user -p pid -o fld -w [cols]
       The traditional switches `-' and whitespace are optional.

DESCRIPTION
       The  top  program provides a dynamic real-time view of a running system.  It can display
system summary information as well as a list of processes or threads currently being managed by the
Linux kernel. The types of system summary information shown and the types, order and size of
information displayed for processes are all user configurable and that configuration can be made
persistent across restarts. The  program  provides  a limited interactive interface for process
manipulation as well as a much more extensive interface for personal configuration  --  encompassing
every aspect of its operation.
       And while top is referred to throughout this document, you are free to name the program
anything you wish.  That new name, possibly an alias, will then be reflected on top's  display  and
used  when reading and writing a configuration file.
```

The UNIX `top` command

Once running, type "u" followed by your username to see solely the processes you are running.

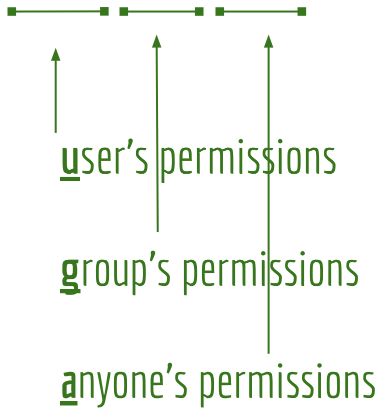
```
top

%Cpu(s):  1.4 us,  1.2 sy,  0.0 ni, 97.3 id,  0.1 wa,  0.0 hi,  0.1 si,  0.0 st
KiB Mem : 65926632 total,  8219192 free, 13037824 used, 44669616 buff/cache
KiB Swap: 16777212 total,  3059908 free, 13717304 used. 44904348 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
28791 u1012898  20   0  115952    712    284 S  28.1  0.0  25:59.39 rsync
28213 u1012898  20   0  146872   5684    960 S   6.6  0.0   6:11.49 sshd
10541 u0105911  20   0  559992  13204   4660 S   3.3  0.0 227:13.23 sview
 3468 dbus      20   0   38032   6440    900 S   2.3  0.0 169:06.49 dbus-daemon
 7868 u6000251  20   0 1128992   5312   3872 S   2.0  0.0 121:29.44 mate-settings-d
20284 u0743456  20   0 7371152 595544 127712 S   2.0  0.9  11:54.73 MATLAB
25653 u1062985  20   0  263692   1884   1684 S   2.0  0.0 164:37.22 vmd_LINUXAMD64
 4369 fastx     20   0 1821332 865596   5324 S   1.3  1.3  40:14.44 node
 7913 u6000251  20   0  430836  37624   2196 S   1.3  0.1  38:55.86 gvfs-udisks2-vo
44738 u6012438  20   0 1135096  12240   3884 S   1.3  0.0  12:34.53 mate-settings-d
    1 root      20   0  293312 107076   1524 S   1.0  0.2  59:25.56 systemd
 3930 u0253283  20   0  439084  38928   2196 S   1.0  0.1  46:47.20 gvfs-udisks2-vo
 4021 u1007787  20   0  157160   3652   1532 R   1.0  0.0   0:00.21 top
 4451 polkitd   20   0  983200 241688   2676 S   1.0  0.4 105:08.02 polkitd
...
```

The UNIX chmod command: change file mode (permissions)

```
NAME
       chmod - change file mode bits
SYNOPSIS
       chmod [OPTION]... MODE[,MODE]... FILE...
       chmod [OPTION]... OCTAL-MODE FILE...
       chmod [OPTION]... --reference=RFILE FILE...
DESCRIPTION
       This  manual  page  documents the GNU version of chmod.  chmod changes the file mode bits of
each given file according to mode, which can be either a symbolic representation of changes to make,
or an octal number representing the bit pattern for the new mode bits.
       The format of a symbolic mode is [ugoa...][[+-=][perms...]...], where perms is either zero or
more letters from the set rwxXst, or a single letter from the set ugo.  Multiple symbolic  modes  can
be given, separated by commas.
       A  combination of the letters ugoa controls which users' access to the file will be changed:
the user who owns it (u), other users in the file's group (g), other users not in the file's group
(o), or all users (a).  If none of these are given, the effect is as if a were given, but bits that
are set in the umask are not affected.
       The operator + causes the selected file mode bits to be added to the existing file mode bits
of each file; - causes them to be removed; and = causes them to be added and causes unmentioned bits
to be removed except that a directory's unmentioned set user and group ID bits are not affected.
```

The UNIX **chmod** command: change file mode (permissions)

```
touch testfile

ls -ltr testfile
-rw-r--r-- 1 u1007787 quinlan 0 Apr 13 06:57 testfile
```



```
r = read privileges
w = write privileges
x = execute privileges
```

The UNIX chmod command: change file mode (permissions)

User	Group	All
rwx	rx	---
111	101	000
7	5	0

The UNIX `chmod` command: change file mode (permissions)

```
chmod 777 my_file
```

The UNIX chmod command: change file mode (permissions)

User	Group	All
rwx	rwx	rwx
111	111	111
7	7	7

The UNIX chmod command: change file mode (permissions)

```
chmod 400 my_file
```

The UNIX `chmod` command: change file mode (permissions)

User	Group	All
`rwx`	`rwx`	`rwx`
`100`	`000`	`000`
`4`	`0`	`0`

You and only you can read the file. For example, SSH keys for Amazon EC2

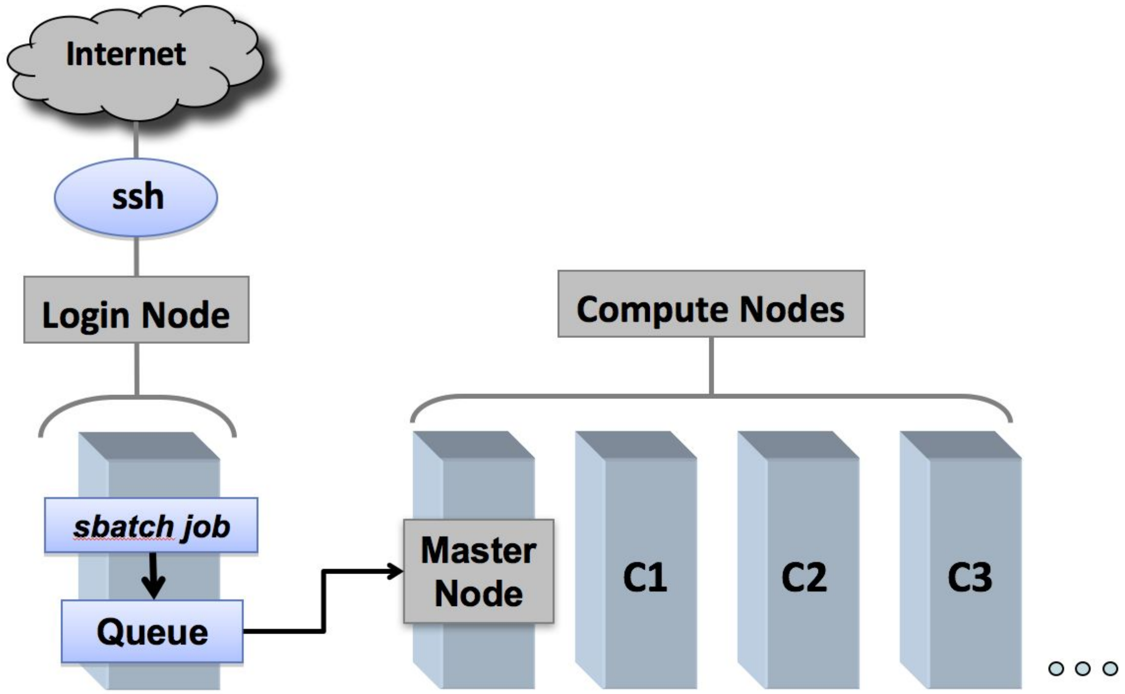
The UNIX **chmod** command: change file mode (permissions)

```
# add write privileges for my group (e.g., lab)
chmod g+w testfile

ls -ltr testfile
-rw-rw-r-- 1 u1007787 quinlan 0 Apr 13 06:57 testfile

# nevermind, I don't trust them
chmod g-w testfile

ls -ltr testfile
-rw-r--r-- 1 u1007787 quinlan 0 Apr 13 06:57 testfile
```

Making a script executable

```
cat sleep.sh
#!/usr/bin/bash
sleep 10

./sleep.sh
bash: sleep.sh: command not found...

ls -l sleep.sh
-rw-r--r-- 1 u1007787 quinlan 25 Apr 13 07:08 sleep.sh

chmod u+x sleep.sh

ls -l sleep.sh
-rwxr--r-- 1 u1007787 quinlan 25 Apr 13 07:08 sleep.sh

./sleep.sh
```

Academic compute clusters



https://cvw.cac.cornell.edu/environment/slurm_intro

Bash script that accepts a sample name and reference genome from the CL

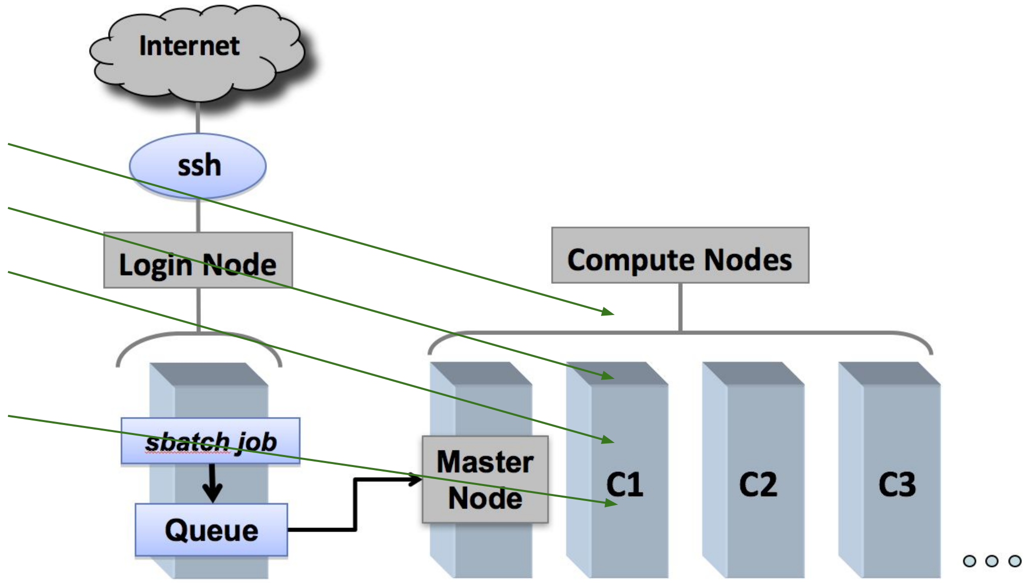
```
vim run2.sh


sample=$1
genome=$2
bwa mem -t 16 $genome $sample.1.fq $sample.2.fq > $sample.sam
samtools view -Sb $sample.sam > $sample.bam
samtools sort -@ 8 -m 1G $sample.bam -o $sample.sorted.bam
samtools index $sample.sorted.bam
freebayes -f $genome $sample.sorted.bam > $sample.vcf

<type Esc then :wq then Enter to save and quit>
```

Run script separately (in parallel) for each sample. **What is the limitation?**

```
bash run2.sh sample1 ref.fa
bash run2.sh sample2 ref.fa
...
bash run2.sh sampleN ref.fa
```

We need to send each job to an available computing resource

```
bash run2.sh sampleA ref.fa
bash run2.sh sampleB ref.fa
bash run2.sh sampleC ref.fa
...
bash run2.sh sampleJ ref.fa
```



We need to make the bash script compatible with SLURM

```
#!/bin/sh
#SBATCH --account=quinlan-kp
#SBATCH --partition=quinlan-kp
#SBATCH -o %j-%N.out        # file to capture STDOUT, job name, Node
#SBATCH -e %j-%N.err        # file to capture STDERR, job name, Node
#SBATCH --time=6:00:00      # expected walltime
#SBATCH --mail-type=FAIL,END
#SBATCH --mail-user=youremail@mail.com
# -----------------------------------------------------------------
sample=$1
genome=$2
bwa mem -t 16 $genome $sample.1.fq $sample.2.fq > $sample.sam
samtools view -Sb $sample.sam > $sample.bam
samtools sort -@ 8 -m 1G $sample.bam -o $sample.sorted.bam
samtools index $sample.sorted.bam
freebayes -f $genome $sample.sorted.bam > $sample.vcf
echo "I am done"
```

https://www.chpc.utah.edu/documentation/software/slurm.php#submit

Submitting jobs to the cluster using SLURM

```
sbatch run2.sh sampleA ref.fa
sbatch run2.sh sampleB ref.fa
sbatch run2.sh sampleC ref.fa
...
sbatch run2.sh sampleJ ref.fa
```

Checking job status

```
# checking all jobs running on cluster
squeue

# checking all of my jobs running on cluster
squeue -u u1007787
JOBID   PARTITION  NAME   USER     ST        TIME  NODES NODELIST(REASON)
2541980 quinlan-k  foo.sh u1007787 PD        0:09      1  kp240

squeue -u u1007787
JOBID   PARTITION  NAME   USER     ST        TIME  NODES NODELIST(REASON)
2541980 quinlan-k  foo.sh u1007787 R        0:02      1  kp240
```

Pending

Running

Checking job status

```
# List all of my jobs stuck in a pending state (queued)

squeue -u u1007787 -t PENDING

# List all of my running jobs

squeue -u u1007787 -t RUNNING

# List detailed information for a job (useful for troubleshooting):

scontrol show jobid -dd <jobid>

# List status info for a currently running job:

sstat --format=AveCPU,AvePages,AveRSS,AveVMSize,JobID -j <jobid> --allsteps
```

Killing a job

```
# checking all of my jobs running on cluster
squeue -u u1007787
   2541952 quinlan-k   foo.sh u1007787 CG       0:01      1 kp244

# oops, I ran the wrong script
scancel 2541952
```

Killing many jobs

```
# checking all of my jobs running on cluster
for jobid in `squeue -u u1007787 | awk '{print $1}' | grep -v
"JOBID"`;
do
    scancel $jobid;
done
```