Analysing target capture data for phylogenomics

Paul Bailey and Alexandre Zuntini

Session 1 - Getting started with the Linux command line

Objectives

- Become familiar with the essential Linux commands
- Handle files and move files around
- Learn about the Kew HPC cluster (HATTA) and how to run programs there
- Use of the Slurm job scheduler

There are plenty of introductory tutorials online - e.g.:

- Programming for Biology 2019
- Command line tutorial for Bioinformatics
- https://unix.t-a-y-l-o-r.com/
- https://learnxinyminutes.com/docs/bash/
- RGB Kew Bioinformatics website has information about the HATTA cluster: https://rbg-kew-bioinformatics-utils.readthedocs.io/en/latest

Logging into HATTA cluster using PuTTY (PC) or iTerm2 (Mac)

```
ssh <user_name>@hatta.ad.kew.org
# or
ssh <user_name>@hatta
```

Easy but essential Linux commands

- List files

```
ls
ls -l
```

```
ls -F
man ls
```

- Creating and changing directories, moving and copying files

```
touch <new_file>
mkdir <dirname>
cd <dirname>
cd ../
cd ~
pwd
mv <my_file> <renamed>
mv <my_file> <../existing_dir>
cp <a_file> <copy_of_file>
cp -pr <dir> <copied_dir>
cp <dir>/<file .
```

- View the contents of a file

```
more of_a_file.txt
tail -n 100 <a_long_file>
cat <file>
vi <file>
```

- Removing files

```
rm <unwanted file>
rmdir <unwanted_empty_dir>
rm -R <unwanted_dir_plus_files>
```

- Other commands and key strokes

  To terminate a program that is running, press and hold down the Ctrl key, then press the C key.

  The up arrow retrieves the previous command. The left and right arrows allow the cursor to move along the command line through existing text.

Any text on the screen (or in the screen buffer) can be selected with the mouse and used on the command line.

Pressing the tab key displays the remaining possibilities for a file path, useful for long file or directory names - e.g.:

`ExamplesData/phylo`

will autocomplete (if a directory or file of this name is present) to:

`ExamplesData/phylogenetic_analysis`

Transferring files to the cluster

- Relevant directories

```
df -h
```

/science/	slow storage, backed up (57TB)
/science/users_area	main users storage area
/data/	fast storage, not backed up, main users work area (88TB)
/home/	home directories (4GB limit)

- CyberDuck can be used to transfer files from your computer to the cluster (easy)
- wget can be useful for downloading data files - e.g.:

```
wget
ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR745/000/SRR7451070/SRR7451070_1.fastq.gz
```

- scp is an alternative way of transferring files

```
scp <source> <target>
# Transfer to the cluster:
scp -p  <file_to_transfer.txt>  <user_name>@hatta:/home/<user_name>/
# Transfer from the cluster:
scp -p  <user_name>@hatta:/home/<user_name>/<file_to_transfer.txt
<target_file_name.txt>
```

The -p flag preserves the timestamp of the transferred file.

- Lots of software is available on GitHib and can be easily downloaded e.g. the HybPiper repository:

```
git clone https://github.com/mossmatters/HybPiper.git
```

Software

Software on the cluster is located in the following areas:
/usr/bin
/usr/local/bin
/data/software
You can install software in your home directory: /home/<user_name>
Most software has a built in help menu when you specify the name of the program on the command line, often requiring a flag to get the full menu - e.g.'s

```
fasttree
blast -h
blast -help
fastatranslate -h
```

The Slurm job scheduler

- Slurm documentation
- Documentation on each Slurm program
- Summary of all Slurm commands
- Information on queues (partitions)

```
sinfo
sinfo -o "%16P %.5a %.10l %8p %.6D %.6t %N"
```

Output columns:
PARTITION - queue name
TIMELIMIT - time limit
PRIO_TIE - priority value

- **Job submission**
  Jobs can be submitted via the sbatch command in one of two ways, either using the sbatch --wrap flag or a Slurm script.

Method 1

```
sbatch -J job_name -p fast -t 5-2:00 -c 2 --mem=100000 \
-o job_name.log -e job_name.log_err --wrap "
command1;  command2;  etc
"
```

Method 2

```
sbatch slurm_script.sh
```

Example of a Slurm script

```
#!/bin/bash
#SBATCH -J job_name
#SBATCH -p fast
#SBATCH -n 1
#SBATCH --mem 400000
#SBATCH -t 6-12:00
#SBATCH -c 1
#SBATCH -o recover_genes.log
#SBATCH -e recover_genes.log
sleep 10
```

Explanation of each parameter flag:
Memory can be specified like this (Default units: megabytes)
--mem=4000 (conventional syntax) or --mem=4G (conventional syntax)
--mem 4000 --mem 4G

-J is your job name
-c is the number of cpus you want to request; cpu are defined here
-p is queue name (see sinfo section above to see other queues)
-t wall time of job - how long you think your job will take to run - reducing this time length helps to make a job start more quickly because the scheduler might be able to fit it in a time slot between other jobs
Acceptable time formats include "minutes", "minutes:seconds", "hours:minutes:seconds", "days-hours", "days-hours:minutes" and "days-hours:minutes:seconds"
e.g. -t 0-2:00 (D-HH:MM) is set for 2 hours

- **To see which of your jobs are running**

```
squeue -u $USER -o "%.18i %.13P %.25j %.8u %.8T %.10M %.9l %.6D %.R %.C %.m %.p"
```

$USER=your user name bash variable
Useful non-default columns:
%.R = tells you what node job is running on
%.C = tells you how many cpus you have specified
%.m = tells you how much memory you have specified (NB - default units are MB)
%.p = how much priority user has

- To see details of a running job

```
scontrol show job=<job_id>
```

- To cancel all pending jobs for a user:

```
scancel -t PENDING -u <username>
```

To cancel a range of jobs

```
scancel {379452..379520}
```

More (Advanced) Unix

- View a list of all previous commands commands

```
history
```

- List equivalent files from any directory with the 'wildcard' character:

```
ls
/science/projects/paftol/PAFTOL_gene_recovery_and_trees/paftools/run1/gene_recover
y/PAFTOL_*-angio353-cds.fasta
```

- Count the number of sequence records in a fasta file:

```
cat /data/projects/training/data/test_targets.fasta | grep '^>' | wc -l
```

This command introduces four new things:

1. passing file contents to another program using a pipe character ('|')
2. using the grep program to search for any text in the input with a regular expression
3. use of characters that have special meaning in a regular expression (i.e. '^')
4. wc is a program that can count words, lines or characters, the -l flag counts the lines

- Search for a file

```
find /science/projects/paftol/AllData/ -name P01-01A_S74_L001_R1_001.fastq.gz -
print
```

- Redirect output to a file from 'stdout' (standard output) using the '>' character. Errors can be sent to standard error ('stderr') by specifying '2' before the '>' character.

```
echo "Redirect some text into a file" > output.txt 2> output.err
```

- Alter read/write/executable permissions for a file - e.g. to make a read only file also executable (some software comes without executable permission)

```
chmod 755 filename
```

Info from 'ls -l' -r--r--r--
read only 4 4 4
Info from 'ls -l' -rwxr-xr-x
read + executable 7 5 5
Summary: r = 4; w = 2; x = 1

- Other useful commands for data processing (mostly); more information can be found in their man pages

```
awk
basename
column
cut
diff
dos2unix
du -ch
export
head
sed
sleep
sort
time
uniq
watch
xargs
zcat
```

Simple Exercise with Slurm

- Create a Slurm job with commands that can perform the following tasks:

1. Count the number of fasta records in this file: /data/projects/training/data/test_targets.fasta
2. Send the program to sleep for 30 seconds ('sleep 30')
3. Use the fastalength program from Exonerate to calculate the length of each sequence record
4. Calculate the sum of all contigs by piping the results from 3. into this command which will print the sum length of all contigs then the average contig length

```
awk '{sum+=$1} END {print "Length of all contigs: " sum "\n" "Average length: "
sum/NR}'
```

5. Use the fastatranslate program to translate the fasta file into protein sequence and redirect the output to a file
6. Once you have submitted the job, view its status in squeue; it should be running!

Hint - you may want to set up the sbatch job and run it first then build up the commands, starting with step 2 would be an easy entry point.

NB - Cluster etiquette - do not run software on the head node!
NB - Cluster etiquette - remember, when you are ready to scale up your analyses, there are only 192 cpus in the current cluster - don't hog them all!

Answer to exercise

```
sbatch -J slurm_test -p fast -t 0-0:15 -c 2 --mem=1000 \
-o job_name.log -e job_name.log_err --wrap "
cat /data/projects/training/data/test_targets.fasta | grep '>' | wc -l
sleep 30
fastalength /data/projects/training/data/test_targets.fasta  | awk '{sum+=\$1} END
{print \"Length of all contigs: \" sum \"\n\" \"Average length: \" sum/NR}'
fastatranslate -F 1 /data/projects/training/data/test_targets.fasta >
test_targets.pep
"
```