Alex's Phoenix Guide

Alex Jackson

March 29, 2016

1 Logging in

In the command line:

```
ssh aXXXXXXX@phoenix.adelaide.edu.au
```

28/3/16 I can connect via the above but not via the below. When I ssh in I think I get into l01 though anyway. It appears that Phoenix only allows you to connect through the university. At the moment, getting around this at home by ssh into dis, then ssh into Phoenix. Will get VPN set up sometime. Okay, VPN is working woopwoop.

is what is says on the wiki but on the email the address is slightly different,

```
ssh a*******@l01.phoenix.adelaide.edu.au
```

2 File transfer

As I'm using linux, I'm going to use terminal commands. This command transfers `myfile.txt` to the desired folder on the Phoenix server. All good here with VPN now, thank god.

```
scp myfile.txt aXXXXXXX@phoenix.adelaide.edu.au:Path/To/MyPhoenixFolder
```

is "secure copy", an extension of the `cp` command.

You can also use "secure file transfer protocol" `sftp`.

```
sftp aXXXXXXX@phoenix.adelaide.edu.au
```

Then you use the `put` and `get` commands to transfer files. I'm not sure how this works yet.

Which one is better? According to `http://www.jscape.com/blog/scp-vs-sftp`:

- SCP is faster
- Both SCP and SFTP use SSH, so have the same level of security
- SFTP has greater functionality (SCP is file transfers *only*)
- They both transfer large files okay
- SFTP can resume file transfers, while SCP cannot.

But since I'll probably just be needing file transfers, SCP is probably the way to go.

3 Loading software

From the wiki: "In most cases, your required software is not loaded by default on the Phoenix system. After logging in, you will need to load your required software before you can perform any calculations. Phoenix uses the `module` system to manage the software environment. To see a list of the available software, use the `module avail` command as in the example below." Tested loading and unloading with R. It works.

Use the commands `module load ModuleName` and `module unload ModuleName` to load and unload the desired software. **We will be using Phoenix to run PSMC only I expect - in comparison, generating data is a shorter process which I should be able to do on the laptop. I'm not sure how**

Phoenix permissions work but we might need to allow some time to negotiate getting PSMC on the Phoenix server.

4 Running scripts

I've got two example job scripts in this folder - `PhoenixScriptExample.sh` (from Lachlan's email) and `PhoenixScriptExampleWiki.sh` (from the wiki). I get the impression from comparing the two, in the `.sh` file you should be able to directly run PSMC. What I'd probably do though is modify Shaun's PSMC loop script. **I might need to do some investigating into "multicore programming". Would it be faster to do lots of PSMC runs at once in parallel, or one at a time and using different cores?**

Included in the job script you need the resources your program is going to use. **I'm not sure how to figure out the number of cores and number of nodes reuired (32 cores per node, Lachlan says in the email that 1 core is usually sufficient so go with that I guess?).** I could probably figure out time by doing a PSMC run (guessing the other Phoenix parameters) and way overestimating the time, and in my job script taking the start and end times. Maybe do this a few times with different genomes. Then that'd give me some idea of how long the script takes. It says it's best to overestimate though, otherwise you can get crashing.

The amount of memory (RAM) allocated is *per node*. **How would I work that out? Is there a way to monitor memory being used if I ran PSMC on the laptop for example, and use that as a guide?**

5 Checking progress

```
squeue -u a*******
```

will list the queue for your login.

6 Queueing jobs

You need to put your script on the queue using `sbatch my_job.sh`. If the script needs variables (mine should be self-contained so unneccessary) see the wikis. After running sbatch, you MUST record the ID since squeue is so long... Also, shouldn't there be like no jobs on my queue since we have our own bit of Phoenix? Maybe "batch" isn't the right place to submit it.

Cancel a job by finding the job ID via `squeue`, and then `scancel obID`.

7 CPU or GPU

From `https://wiki.adelaide.edu.au/hpc/index.php/CPU_tasks_VS._GPU_tasks`: "If your research involves large matrices, intensive numerical and mathematical computations, or is image intensive, there is a good chance that you should consider paralleling computing. If you are in doubt, again, please do not hesitate, give the Phoenix team a call or email us, we are here to support your research."

PSMC is computationally intensive, right? So it might be worth it to try out GPU processing, or at least contact the Phoenix team.

Running a GPU script is slightly different to a CPU (normal) script.

```
#!/bin/bash

# Configure the resources required
#SBATCH -p batch                                        # partition (this is the queue your job
    will be added to)
#SBATCH -n 8                                             # number of cores (here uses 8, up to 32
    cores are permitted)
#SBATCH --time=01:00:00                                  # time allocation, which has the format
    (D-HH:MM), here set to 1 hour
#SBATCH --gres=gpu:4                                     # generic resource required (here
    requires 4 gpu cores)
```

```
#SBATCH --mem=16GB                                      # memory pool for all cores (here set to
    16 GB)

# Configure notifications
#SBATCH --mail-type=END                                 # Type of email notifications will be
    sent (here set to END, which means an email will be sent when the job is done)
#SBATCH --mail-type=FAIL                                # Type of email notifications will be
    sent (here set to FAIL, which means an email will be sent when the job is fail to complete)
#SBATCH --mail-user=my_email@adelaide.edu.au            # Email to which notification will be
    sent

# Execute your script (due to sequential nature, please select proper compiler as your script
    corresponds to)
bash ./my_program.sh
```