Notes on CUDA practicals on TACC's Lonestar6 system

Prof. Mike Giles
`mike.giles@maths.ox.ac.uk`

1 The Lonestar6 system

The layout of the A100 GPU nodes on the Lonestar6 system is shown in Figure 1. The GPU nodes on Lonestar6 each have 3 A100 GPUs but you will be using only one of them in the practicals.

The login node (which is actually a set of nodes) sits on the external network and can be accessed from machines on the university network through an SSH command:

```
ssh -X <username>@ls6.tacc.utexas.edu
```

Note that your username on TACC may be different to your username on your laptop or other home system which is why you will probably need to specify it.

The `-X` option provides X-window forwarding which is convenient when using windows-based editors such as `emacs`, or the Nsight IDE. Alternatively, TACC recommends the use of either DCV or VNC for remote viewing.

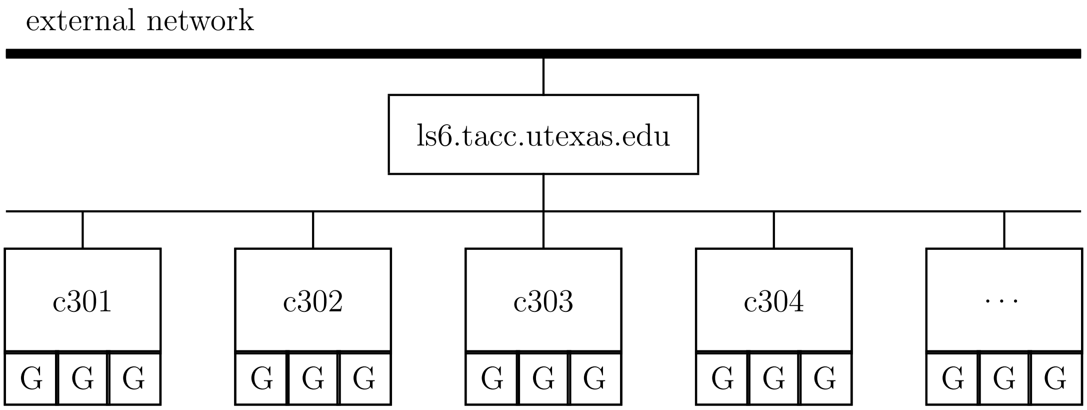

Figure 1: TACC's Lonestar6 system with login node(s) and compute nodes

2 Practical files

The tar file `practicals.tar.gz` can be downloaded from https://people.maths.ox.ac.uk/gilesm/cuda/practicals.tar.gz and put in your working directory using the commands:

```
cd $WORK
wget https://people.maths.ox.ac.uk/cuda/practicals.tar.gz
```

It can then be unpacked using the command:

```
tar zxvf practicals.tar.gz
```

which will give you a separate sub-directory for each practical plus a few header files in the top-level directory.

3 CUDA module

To use the CUDA compiler and other CUDA tools you need to add the CUDA module using the command:

```
module add cuda
```

which will load the default version of the CUDA module which is currently 11.4. The command:

```
module save
```

will ensure that the module is automatically reloaded the next time you login.

Some of the later practicals also require you to set the following environment variable so that Make can correctly find certain CUDA library header files:

```
export CUDA_HOME=/opt/apps/cuda/11.4/
```

4 Interactive sessions on the GPU nodes

Editing, and perhaps also compiling, is best done on the login node, but to run your CUDA codes you need to create an interactive session on one of the GPU nodes by issuing the following command:

```
idev -p gpu-a100-shared
```

You can then compile your code as usual, and run it on one of the GPUs attached to the compute node. Using an interactive session like this can be appropriate when developing codes using test runs which last no more than a few seconds, but for anyone doing "production" runs you should probably use the Slurm batch scheduler.