Preliminary

It is assumed that you have the basic knowledge of programming in Linux environment. If you have difficulty in starting to program in Linux, please let us know. It is also expected that you have already been instructed on how to login to the computer cluster that are used throughout this workshop, the Fornax cluster.

These exercises are modified after the Fornax documentation in http://www.ivec.org/services/supercomputing/documentation/fornaxuwa/pbspro, in order to summarize the process for you to execute your programs. Also in the process, we will learn about the hardware, especially the GPUs.

Part 1: Modules in Fornax Cluster

Fornax cluster is installed with Linux distro, Centos 6.2. This means that you will have access to all the Linux commands such as `less`, `cat`, `vim`, `cp`, `rm` and so on. In the default directory, find the file *helloworld.c* which you can probably guess that it is a simple *hello world* program in C.

Programming in a cluster environment is just slightly different from programming in a desktop environment. In Fornax, you need to load the specific modules in order to do anything.

Firstly you will need to load the gcc compiler and openmpi (needed for execution of the program) by the command

```
module load gcc

module load openmpi
```

Then compile the program with

```
gcc helloworld.c
```

You can look through the available modules with the command

```
module avail
```

We will look at how to execute the program in the next part.

Part 2: Basic Job Script

In a cluster environment, programs are usually executed in a queue. In our case, we need to use a job script to submit our programs to a queue. An example job script is given below:

```
#!/bin/bash
#PBS -W group_list=projectname
#PBS -q courseq
#PBS -l walltime=00:05:00
#PBS -l select=1:ncpus=1:ngpus=1:mem=1gb
#PBS -l place=excl

module load gcc
module load openmpi

mpirun ./programname
```

Line by line, in the above script,

```
#!/bin/bash
```

This specifies that the file is written for the Bourne Again SHell (BASH).

```
#PBS -W group_list=projectname
```

Everyone will be assigned a *projectname,* which you need to put it in this line. Your current project name is *courses01.*

```
#PBS -q courseq
```

This specify the name of the queue you are going to submit your jobs.

```
#PBS -l walltime=00:05:00
```

This line specifies the amount of time you expect the job to run for. For example, the above line is requesting 5 minutes.

```
#PBS -l select=1:ncpus=1:ngpus=1:mem=1gb
```

This specifies the number of nodes and resources per node your job requires. In the above example, 1 node is requested, with 1 cpu **core** per node, with 1 GPU, and 1 GB of memory. Note that ncpus refers to the number of cores, rather than the physical number of CPUs.

```
#PBS -l place=excl
```

This line requests the exclusive use of the node’s resources.

Note that you can also do the modules loading in the job script rather in the terminal.

Part 3: Work Queue

After creating your job script, inserting your program name which by default is named *a.out* if you did not use the `–o` option, you can submit your job script by the command

```
qsub jobscriptname
```

For checking the queue status, you can use the command

```
qstat
```

With your *hello world* program, you probably cannot see it in a queue, as it is such a small program. There you need to use the command

```
qstat –H –u username
```

Warning: Do not use `–H` option without `–u` as you will get all the previously completed jobs since the beginning of time (in Fornax).

You can also check out the status of the queues with the command

```
qstat -Q
```

More details of the options can be found in the manual

```
man qsub

man qstat
```

Part 4: Execute Pre-compiled Programs

Some loadable modules in the cluster might include some pre-compiled programs that you can execute. We will look at the CUDA SDK which can tell us about the GPUs equipped in the cluster.

First load the CUDA and SDK module

```
module load cuda

module load cuda-sdk
```

Then modify your job script to run a program name called *deviceQuery*. You can read the output generated, it tells you a lot about the GPU that is equipped in the nodes. These numbers will become crucial when you start optimizing your CUDA program.

For more executable programs from the SDK, type

```
which deviceQuery
```

to find out the directory of the SDK, then use the `ls` command to see what are the available programs. Note that some programs might require a graphics output and cannot be executed in Fornax (such as oceanFFT).

Part 5: Interactive PBS Session

It is not always desirable to submit our programs to a queue. For instance, when we want to run many small jobs for debugging purposes, we would like to check the output immediately.

So we have the option to use a node exclusively, by entering an *Interactive PBS Session*. This can be done by doing

```
qsub -I -W group_list=projectname
```

This will enable you to use one of the nodes just like a desktop computer. You can run your program with the command

```
./a.out
```

Note that this session will expire after a certain time, and you will get kicked out from the session. When that happens, just re-login with the same command.

This concludes the exercises about programming in Fornax cluster. Hope you enjoy the rest of the workshop.