Introduction to the KNL Exercise

Adrian Jackson

24 October 2016

1 Introduction

In this exercise we are going to log on to the ARCHER KNL system and run a simple program.

2 Using the system

For this tutorial we will be using the ARCHER CrayXC40 KNL system. You should have an account on this system already. To access the KNL system you need to ssh to:

```
ssh login.archer.ac.uk
```

Once on these login nodes you need to ssh to the KNL system itself (if you want to open any windows, i.e. emacs, you'll need to add a –Y or –X flag to the command below):

```
ssh knl-login
```

From here you can compile and submit jobs. It is often useful to know how the KNLs in the system are setup. Cray provides a command that will enable querying the KNL setup:

```
apstat -M
```

This will tell you the cluster mode of each of the KNLs and how much MCDRAM is available, and how much MCDRAM is being used for cache.

To submit jobs to the system we will be using the PBS batch system. Below is an example of a SLURM batch script we can use to run a job:

```
#!/bin/bash
#PBS -N numactljob
#PBS -l select=1:aoe=quad_100
#PBS -l walltime=00:10:00
#PBS -A BUDGETCODE

cd $PBS_O_WORKDIR
aprun -n 1 numactl -H
```

The line `#PBS -l select=1:aoe=quad_100` specifies the setup of the KNL required by the job. If this is not specified the default will be Quad Cache mode (cluster mode in quadrant setup, MCDRAM in cache mode). The cluster is configured with some nodes in quad cache mode and some in quad flat mode. It is also possible to re-configure nodes to other modes, but we won't need to do with for these practicals. If you want to run on quad flat modes you should change the `aoe` line to `#PBS -l select=1:aoe=quad_0`.

The `#PBS -A BUDGETCODE` line specifies what budget your job should be charged to (as with the main ARCHER system). By default, on the KNL system, you will have a budget named `knl-$user` (where `$user` is your username on the KNL system, and you need to put that budget name in place of `BUDGETCODE` in the above script for it run.

To run a job on the system we use the `qsub` command, i.e. (assuming the script above is called runnumactl.sh)

```
qsub runnumactl.sh
```

You can `qstat` to see running jobs (`qstat -u $USER` will show only your jobs) and `qdel` to cancel a job.

The `aprun` command in the script above is the Cray job launcher which runs the executable on the KNL system. aprun takes a number of different arguments, i.e.:

```
aprun -n 1 -d $OMP_NUM_THREADS -j 4 -cc depth ./myexecutable
```

Here `-n 1` specifies the number of processes (or copies of you executable) to run. `-d` specifies how many threads each process will launch. `-j` specifies how many hyperthreads to use per physical core. `-cc` specifies how to bind threads to physical cores. The exercises we are doing in this practical will have submission scripts with aprun already specified in them.

We are using the Cray compiler setup on this machine. This means the C compiler is called `cc` and the Fortran compiler is called `ftn`, with the specific compile defined by the program environment module that is loaded. By default the Cray compilers (`PrgEnv-cray`) are loaded when you log in to the system. For the practicals we want to use the Intel compiler so you need to run the following command to switch to those compilers:

```
module swap PrgEnv-cray PrgEnv-intel
```

On the ARCHER KNL system we have the same type of file systems that we have on the main ARCHER system, and /home directory which is backed up and used for compilation etc… (in fact it's the same /home filesystem as you use on the main machine) and a /work filesystem that is used when running jobs. As with the main ARCHER system, /home is not visible from the compute nodes (the KNLs), so applications and datasets need to be on /work for anything you submit to the compute nodes. Your /work directory will be:

```
/work/knl-users/$USER
```

3 Running a basic program

To get started on the system download the following file (you can use `wget` on the main ARCHER login nodes, no the knl login nodes, to do this):

https://www.archer.ac.uk/training/course-material/2016/11/161101_KNL_EPCC/Exercises/MemorySource.tar.gz

You should be able unpack it with the command `tar zxvf MemorySource.tar.gz`. It may be sensible to do this on the /work file system rather than /home.

For now we will only use `xthi.c`, in `MemorySource/xthi` which we will use to investigate thread placement on the KNL. Compile xthi and run it using the submission script. Alter the aprun parameters to run with different numbers of MPI processes and OpenMP threads.

Try running on more than one KNL. To do this you'll need to change the `#PBS -lselect=1` line to specify the number of nodes (KNLs) you want to use, and also the aprun line, adding a `-N` flag to specific the number of MPI processes per node. You can use at most 8 KNLs at any one time.