PCSE Lecture 2

Hello World!

Cyrus Proctor
cproctor@tacc.utexas.edu

January 22, 2015

Let's Get Started!

- Login to the lab computers using your UT EID and password
- Open up a browser session and navigate to TACC's Homepage
- Navigate to the Stampede User Guide
- On the desktop, open up Putty
- Open two terminal sessions

Login to Stampede

ssh taccusername@stampede.tacc.utexas.edu
Enter your password you used for your TACC portal account.
Have two ssh sessions to Stampede started

We Might be Able to Use Our VM's

- Login to the lab computers using your UT EID and password
- Go to http://tacc-facvdi.austin.utexas.edu
- You will see a screen with a single computer icon labeled Centos 6, click on it
- It will download a file. Click on the file to open it, it should open the connector software that will take you to a blue login page for the Linux instance. Username student, password is hookem.
- Once you click through that screen you should be staring at a vanilla Centos install.
- Open two terminal sessions

Login to Stampede

ssh -Y taccusername@stampede.tacc.utexas.edu
Enter your password you used for your TACC portal account.
Have two ssh sessions to Stampede started

Modules

- On Stampede and most TACC computers we use Lmod
- Modules add information, usually, to your $PATH and $LD_LIBRARY_PATH environment variables
- Software management system that keeps track of software stacks
- Lmod commands
  - module list (ml for shorthand)
  - module purge (ml purge)
  - module reset (ml reset)
  - module load banana (ml banana)
  - module swap banana monkey (ml monkey)
  - module remove banana (ml rm banana)
  - module spider banana (ml spider banana)
  - module help banana (ml help banana)

Before we continue

ml reset
ml intel/14.0.1.106
ml impi

Stampede File Systems

- $HOME
  - This directory has a quota limit of of 5GB, 150K files
  - This file system is backed up.
  - Use "cdh" as an alias
- $WORK
  - This directory has a quota limit of of 1TB, 3M files
  - This file system is not backed up
  - Use "cdw" as an alias
- $SCRATCH
  - Change to this dir in your batch scripts and run jobs on here
  - Purge Policy: Files with access times greater than 10 days are purged
  - Use "cds" as an alias
- Also /tmp and $ARCHIVE

Let's Do Some Work

For both of your ssh terminals:

- cdh
- mkdir pcse_2015
- mkdir pcse_2015/hello
- mkdir pcse_2015/hello/serial
- cd pcse_2015/hello/serial

Interactive Development Session (idev)

For one window, stay on login node.
For the other, let's start an idev session
idev -m 120 -A PCSE-2015

Text Editors

vim

- vim filename
- i to enter "i"nsert mode
- esc to go back to command mode
- :w in command mode to write to a file
- :wq in command mode to write and quit

Emacs

- emacs filename
- C-x C-s to save buffer into filename
- C-x C-c to save buffer and quit

Serial Hello World!

Fortran

```
program hello
    implicit none

    write(*,*)"Hello World!"

end program hello
```

C

```
#include <stdio.h>

int main(int argc, char*
    argv[]){

  printf("Hello World!\n");

  return 0;

}
```

```
ifort -o hello_f hello.F90
icc   -o hello_c hello.c
```

OpenMP Hello World!

Fortran

```
program hello_omp
  use omp_lib
  implicit none

  integer :: thread_id, num_threads

  !$omp parallel private(thread_id)

    num_threads = omp_get_num_threads()
    thread_id = omp_get_thread_num()

    write(*,*)"Hello World! I am ", thread_id, " out of ",
         num_threads

  !$omp end parallel

end program hello_omp
```

```
ifort -openmp -o hello_omp_f hello_omp.F90
icc   -openmp -o hello_omp_c hello_omp.c
```

OpenMP Hello World!

C

```c
#include <omp.h>
#include <stdio.h>

int main (int argc, char *argv[]){

  int thread_id, num_threads;

  #pragma omp parallel private(thread_id)
  {
    num_threads = omp_get_num_threads();
    thread_id = omp_get_thread_num();

    printf("Hello World! I am %d of %d\n", thread_id, num_threads);
  }

  return 0;
}
```

```
ifort -openmp -o hello_omp_f hello_omp.F90
icc   -openmp -o hello_omp_c hello_omp.c
```

Batch Submit

```
#!/bin/bash
#SBATCH -J hello              # job name
#SBATCH -o hello.o.%j         # output file name (%j expands to jobID)
#SBATCH -e hello.o.%j         # error file name  (%j expands to jobID)
#SBATCH -n 1                  # total number of mpi tasks requested
#SBATCH -N 1                  # total number of Nodes requested
#SBATCH -p normal-mic         # queue (partition) -- normal-mic, etc.
#SBATCH -t 00:05:00           # run time (hh:mm:ss)
#SBATCH -A PCSE-2015          # Account name to charge SU's to.


export OMP_NUM_THREADS=1
export MIC_OMP_NUM_THREADS=1

ibrun ./a.out
```

To Submit Batch Job

From the login node:

sbatch batch_script

Then, to watch the job:

watch squeue -u username

MPI Hello World!

Fortran

```
program hello_mpi
  use mpi
  implicit none

  integer :: ierr, my_task_id, num_tasks

  call MPI_init(ierr)


  call MPI_Comm_rank(MPI_COMM_WORLD, my_task_id, ierr)
  call MPI_Comm_size(MPI_COMM_WORLD, num_tasks,  ierr)

  write(*,*)"Hello World! I am ", my_task_id, " out of ", num_tasks

  call MPI_Finalize(ierr)

end program hello_mpi
```

```
mpif90 -o hello_mpi_f hello_mpi.F90
mpicc  -o hello_mpi_c hello_mpi.c
```

MPI Hello World!

C

```c
#include <stdio.h>
#include "mpi.h"

int main(int argc, char* argv[]){

  int ierr, my_task_id, num_tasks;

  ierr = MPI_Init(&argc, &argv);


  MPI_Comm_rank(MPI_COMM_WORLD, &my_task_id);
  MPI_Comm_size(MPI_COMM_WORLD, &num_tasks);

  printf("Hello World! I am %d out of %d\n", my_task_id, num_tasks);

  MPI_Finalize();

  return 0;
}
```

```
mpif90 -o hello_mpi_f hello_mpi.F90
mpicc  -o hello_mpi_c hello_mpi.c
```

Hybrid Hello World!

Fortran

```
program hello_hybrid
  use mpi
  use omp_lib
  implicit none

  integer :: ierr, my_task_id, num_mpi_tasks
  integer :: thread_id, num_omp_threads

  call mpi_init(ierr)
  call mpi_comm_rank(MPI_COMM_WORLD, my_task_id, ierr)
  call mpi_comm_size(MPI_COMM_WORLD, num_mpi_tasks,  ierr)

  !$omp parallel private(thread_id)
    num_omp_threads = omp_get_num_threads()
    thread_id = omp_get_thread_num()
    write(*,"(4(a,i3),a)") &
              "Thread ", thread_id, &
              " out of ", num_omp_threads, &
              " OpenMP threads; MPI task ", my_task_id, &
              " out of ", num_mpi_tasks, " MPI tasks."
  !$omp end parallel

  call mpi_finalize(ierr)
end program hello_hybrid
```

```
mpif90 -openmp -o hello_hybrid_f hello_hybrid.F90
mpicc  -openmp -o hello_hybrid_c hello_hybrid.c
```

Hybrid Hello World!

C

```c
#include <stdio.h>
#include "mpi.h"
int main(int argc, char* argv[]){

  int ierr, my_task_id, num_mpi_tasks;
  int thread_id, num_omp_threads;

  ierr = MPI_Init(&argc, &argv);
  MPI_Comm_rank(MPI_COMM_WORLD, &my_task_id);
  MPI_Comm_size(MPI_COMM_WORLD, &num_mpi_tasks);

  #pragma omp parallel private(thread_id)
  {
    num_omp_threads = omp_get_num_threads();
    thread_id = omp_get_thread_num();

    printf("Thread %d out of %d OpenMP threads; MPI task %d out of %d MPI tasks.\n",
           thread_id, num_omp_threads, my_task_id, num_mpi_tasks);
  }
  MPI_Finalize();
  return 0;
}
```

```
mpif90 -openmp -o hello_hybrid_f hello_hybrid.F90
mpicc  -openmp -o hello_hybrid_c hello_hybrid.c
```

MPI Symmetric Hello World!

Fortran

```
program hello_mpi
  use mpi
  implicit none

  integer :: ierr, my_task_id, num_tasks
  logical :: on_mic

  call MPI_init(ierr)
  call MPI_Comm_rank(MPI_COMM_WORLD, my_task_id, ierr)
  call MPI_Comm_size(MPI_COMM_WORLD, num_tasks,  ierr)

  on_mic = .false.
#ifdef __MIC__
  on_mic = .true.
#endif

  if (on_mic) then
    write(*,*)"On the MIC!  I am ", my_task_id, " out of ", num_tasks
  else
    write(*,*)"On the HOST! I am ", my_task_id, " out of ", num_tasks
  end if
  call MPI_Finalize(ierr)
end program hello_mpi
```

```
mpif90 -xhost -o hello_symmetric_mpi_f.cpu hello_symmetric_mpi.F90
mpif90 -mmic  -o hello_symmetric_mpi_f.mic hello_symmetric_mpi.F90
mpicc  -xhost -o hello_symmetric_mpi_c.cpu hello_symmetric_mpi.c
mpicc  -mmic  -o hello_symmetric_mpi_c.mic hello_symmetric_mpi.c
```

MPI Symmetric Hello World!

C

```c
#include <stdio.h>
#include "mpi.h"

int main(int argc, char* argv[]){

  int ierr, my_task_id, num_tasks;
  int on_mic;
  ierr = MPI_Init(&argc, &argv);
  MPI_Comm_rank(MPI_COMM_WORLD, &my_task_id);
  MPI_Comm_size(MPI_COMM_WORLD, &num_tasks);

  on_mic = 0; // false
#ifdef __MIC__
  on_mic = 1; // true
#endif

  if(on_mic) {
    printf("On MIC!  I am %d out of %d\n", my_task_id, num_tasks);
  }
  else {
    printf("On HOST! I am %d out of %d\n", my_task_id, num_tasks);
  }
  MPI_Finalize();
  return 0;
}
```

```
mpif90 -xhost -o hello_symmetric_mpi_f.cpu hello_symmetric_mpi.F90
mpif90 -mmic  -o hello_symmetric_mpi_f.mic hello_symmetric_mpi.F90
mpicc  -xhost -o hello_symmetric_mpi_c.cpu hello_symmetric_mpi.c
mpicc  -mmic  -o hello_symmetric_mpi_c.mic hello_symmetric_mpi.c
```

OMP Offload Hello World!

Fortran

```
program hello_omp
  use omp_lib
  implicit none
  integer :: thread_id, num_threads
  logical :: on_mic
  on_mic = .false.

  !$omp target
    !$omp parallel private(thread_id)

#ifdef __MIC__
      on_mic = .true.
#endif
      num_threads = omp_get_num_threads()
      thread_id = omp_get_thread_num()
      if (on_mic) then
        write(*,*)"On the MIC!  I am ", thread_id, " out of ", num_threads
      else
        write(*,*)"On the HOST! I am ", thread_id, " out of ", num_threads
      end if
    !$omp end parallel
  !$omp end target

end program hello_omp
```

```
ifort -openmp -o hello_offload_omp_f hello_offload_omp.F90
icc   -openmp -o hello_offload_omp_c hello_offload_omp.c
```

OMP Offload Hello World!

C

```c
#include <omp.h>
#include <stdio.h>
int main (int argc, char *argv[]){
  int thread_id, num_threads;
  int on_mic;
  on_mic = 0; // False

  #pragma omp target
  {
    #pragma omp parallel private(thread_id)
    {
#ifdef __MIC__
      on_mic = 1; // true
#endif
      num_threads = omp_get_num_threads();
      thread_id = omp_get_thread_num();
      if (on_mic) {
         printf("On MIC!  I am %d of %d\n", thread_id, num_threads);
       }
       else {
         printf("On HOST! I am %d of %d\n", thread_id, num_threads);
      }
    }
  }
  return 0;
}
```

```
ifort -openmp -o hello_offload_omp_f hello_offload_omp.F90
icc   -openmp -o hello_offload_omp_c hello_offload_omp.c
```

Summary

- Utilize the TACC User Guide
- Login to Stampede via ssh; use -X or -Y for X11 tunneling
- Take advantage of the Lmod module framework
- Understand and respect common user space on login nodes
- Compile jobs on login nodes or in development queue
- Run jobs with idev session or with batch submit