Using Your Blue Waters Accounts

Introduction to HPC

from the Fall 2016 Team

With thanks to Bill Gropp and the Spring 2016 Team

Logging In

- Each account has a user name traxxx, e.g., tra798
- Each student should use only their account, and only for the course
- Passwords can*not* be changed
- Access Blue Waters through a special gateway (do not use the h2ologin.ncsa.illinois.edu node mentioned in the online documentation)
- E.g.,

  ssh tra798@bwbay.ncsa.illinois.edu

Development Environment

- Blue Waters runs a version of Linux
  - ♦ All "usual" Linux commands available
- "module" used to select different software, including compilers
- Jobs are run by submitting a script file to a *batch system* with the command **qsub**. qstat shows the status of the Blue Water job queue.

Debugging

- Interactive debugging is possible but somewhat complicated
  - See https://bluewaters.ncsa.illinois.edu/ddt
- It is better (as much as possible) to debug on smaller systems (I use my laptop for most of my MPI debugging) before running on Blue Waters.

Moving Files to and From Blue Waters

- See
  - https://bluewaters.ncsa.illinois.edu/education-training-allocation-data-transfer
- scp (secure copy program) is an easy way to move files between different systems
  - For large file transfers (bigger than you'll need for the homework), see
  - https://bluewaters.ncsa.illinois.edu/data-transfer-doc
- You (probably) can't use scp *to* Blue Waters
  - Due to the way security is set up for these accounts
- The easiest solution is to log into Blue Waters and use scp *from* Blue Waters
- For groups of files, create a compressed tar file first and move that.

Running Programs on Blue Waters

- See the documentation at https://bluewaters.ncsa.illinois.edu/
- Particularly the "Documentation/Getting Started"
- Note that jobs should *not* be run on the login node
  - ♦ Use only for editing, compiling, linking, etc.
  - ♦ Use the batch system for all computational experiments, even with a single node

Other Notes

- For information on how to control how nodes are allocated, see https://bluewaters.ncsa.illinois.edu/topology-considerations
- MPI jobs are run with aprun, not mpiexec
  - An mpiexec is provided that will submit batch jobs, run the program with aprun, and direct the output to stdout/stderr. This is useful for simple tests. Details on a later slide

Fixing Your Environment

- By default, vim uses a nearly useless color palate for syntax highlighting, making it nearly impossible to read
  - Add
    - syntax off
  - to the file .vimrc :
    - echo “syntax off” >> .vimrc

More on Running Programs

- Create a script, run qsub with that script, and then wait for the job to finish
- Example follows, but
  - "Account" for the class is bagx
  - Use this with the PBS –A command

An Example Script File for ps2.c

```
#!/bin/bash
#PBS -q normal
#PBS –A bagx
#PBS -N ps2
# Always request the entire node.  ppn is now the
# processors per node, which is 32 for xe and 16 for xk
#PBS -l nodes=1:ppn=32:xe
#PBS -l walltime=0:05:00
#PBS -e $PBS_JOBID.err
#PBS -o $PBS_JOBID.out
cd $PBS_O_WORKDIR
# See the man page on aprun
aprun -n 1 -N 1 -cc 0 ps2.out –da_refine 4 –log_view
```

Running ps2

- cp –iR /projects/eot/bagx/ps2 ./
- cd ps2/mod/
- module load cray-petsc
- module swap PrgEnv-cray PrgEnv-gnu
- make all
- qsub test.pbs