Using the Argo Cluster

Paul Sexton
CS 566
February 6, 2006
Logging in

- The address is argo-new.cc.uic.edu
- Use ssh to log in, using your ACCC username
- Ex: ssh psext02@argo-new.cc.uic.edu
- Pay attention to the login message, this is how the admins will communicate information to you.
Environmental Variables

Add the following to your .bash_profile:

```
# For the PGI compilers
export PGI_HOME=/usr/common/sgi-6.0-0/linux86/6.0
export LIB=$PGI_HOME/lib:$PGI_HOME/liblf:/lib:/usr/lib

# For the MPI libraries
export MPICH2_HOME=/usr/common/mpich2-1.0.1
export LD_LIBRARY_PATH=$MPICH2_HOME/lib:$LD_LIBRARY_PATH
export PATH=$MPICH2_HOME/bin:$PATH
```
Compiling with GCC

- MPICH provides wrapper scripts for C, C++, and Fortran.
- Use mpicc in place of gcc
- Use mpicxx in place of g++
- Use mpif77 in place of g77.
Compiling with PGI

- Use pgcc in place of cc/gcc
- Use pgCC in place of CC/g++
- Use pgf77 in place of f77/g77.
- No wrapper scripts provided, need to add include and lib paths manually:
Compiling with PGI (cont.)

- For C or C++, add the following to your compiler flags:
  - `-I$MPICH2_HOME/include`
  - `-L$MPICH2_HOME/lib`
  - `-lmpich`

- For Fortran, also add:
  - `-lfmpich`
Running an MPI program

- Normally, to run an MPI program, you’d use the mpirun/mpiexec command.
- Ex: mpirun -np 4 myprogram
- Don’t do this on argo.
Running an MPI program on Argo

- Edit and compile your code on the master node, and run it on the slave nodes.
- Using PBS / Torque.
- The important commands:
  - qsub
  - qdel
  - qstat
  - qnodes
qsub - submit a job

- qsub [options] script_file
- script_file is a shell script that gives the path to your executable and the command to run it.
- Do not call qsub on your program directly!
- Returns a status message giving you the job id.
  - Ex: “3208.argo-new.cc.uic.edu”
**A qsub script**

- For example, imagine your program is called “foo.exe”.
- The contents of the script file would be:
  ```bash
  mpiexec -n 4 /path/to/foo/foo.exe
  ```
- Where “/path/to/foo/” is the path to the executable’s directory.
**qdel - remove a job**

- `qdel job_id`
- `job_id` is the number `qsub` returned
- Ex: `qdel 3208`
qstat - queue status

- `qstat job_id`
  - To check on the status of a job when you know the number
  - Ex: `qstat 3208`

- `qstat -u userid`
  - To check on the status of all of your jobs.
  - Ex: `qstat -u psexto2`

- `qstat`
  - To obtain status of all jobs running on argo.
qnodes - node status

- Shows the availability of all 64 nodes, and how many jobs they’re currently running.
- No arguments.
- No man page entry for some reason.
Viewing the output

- Standard output and standard error are both redirected to files:
- [script name].{e,o}[job id]
- Ex:
  - stderr in my_script.e3208
  - stdout in my_script.o3208
- The .e file should usually be empty.
More on qsub

- There are numerous options to qsub, but the most common / useful one is “-l nodes”:

- To set the number of nodes:
  - qsub -l nodes=4 my_script

- To pick the nodes manually:
  - qsub -l nodes=argo9-1+argo9-2+argo9-3 my_script

- To run only on a subset of nodes:
  - qsub -l nodes=6:cpu.xeon my_script
  - qsub -l nodes=8:cpu.amd my_script
  - qsub -l nodes=8:cpu.smp my_script
And that’s about all there is...

Final Notes:

- The Intel compilers are also available.
- The PGI tools include a debugger, PGDBG, and a profiler, PGPROF. These are highly recommended for debugging and profiling parallel code.
- Limit yourself to 3 jobs at a time.