Grid 101

Josh Hegie

Overview

- How to access the Grid
- Working on the Grid
- Running jobs with SGE
- How to compile MPI
- Running MPI on the GRID

Accessing the Grid

- Hostname is: hnode.research.unr.edu
- Username: netID
- Password: netID password

Accessing the Grid

- hnode
  - The frontend and login server
  - Write your code here, but consider compiling elsewhere (more on that later)
  - Do NOT run your jobs here
    - Everyone has to log in and work here, so be nice

Working on the Grid

- You can access a node by typing:
  - qrsh –q ECSL.owner
  - This will give you a shell on a remote machine

Running Jobs

- The UNR Grid uses the Sun Grid Engine (SGE)
  - Provides several useful commands
    - qsub
    - qstat
    - qdel

qsub

- Submits a job to a queue
- Typically a shell script (I use bash)
- qsub options begin with #$
  - Most of these are completely optional

qsub Options

- -cwd
  - The default drops output in ~/
  - This moves it to the current directory
- -N <name>
  - Normally your job is named based on the script name, this changes it whatever you specify
- -pe <PE Name> <n>
  - This determines how nodes are allocated
  - More on this later

qsub Options

- -M email@place.com
  - Where to send mail to
- -m bes
  - Send mail when the job (b)egins, (e)nds or is (s)uspended
- -q ECSL.owner
  - Send to the ECSL queue, as opposed to the common queue
- -l mem_total=3G
  - Not necessary, requests a machine with at LEAST 3 GB of memory, if there is no such machine, the job may never get scheduled
- If you want to know more, then read the man pages

qstat

- qstat is used to get information about queued and running jobs
- Just calling “qstat” will show your jobs
  - if you instead use qstat –u ‘*’ you can see jobs for everyone

qstat

- Occasionally jobs fail and error out
  - qstat –j <job-id> gives information about a specific job, including any errors in scheduling
  - qstat –t gives information about which nodes are in use and which are the masters and the slaves

qdel

- Occasionally you need to remove a job, when this happens you need qdel
- Invoked as: qdel <job-id>

Sample Submit Script

```
#!/bin/bash
#$ -cwd
#$ -pe SharedMem 4
#$ -N pe_run1
#$ -M jhegie
#$ -m bes
#$ -q ECSL.owner

echo "Hello World!”
echo -n "I've allocated $NSLOTS CPU cores on ”
echo -n $(cat $TMP/machines)
echo " "
```

Sample Submit Script

- This creates job with 4 nodes on one machine and echoes “Hello World!” and lists that machine
  - It will wait for a single machine with 4 nodes to be available

Sample Submit Script

- You'll notice some files in your directory now
  - pe_run1.o<job-ID> is your program's stdout
  - pe_run1.e<job-ID> is your program's stderr
  - pe_run1.po<job-ID> is output from the parallel environment
  - pe_run1.pe<job-ID> is errors from the parallel environment
    - The p files should be empty

Compiling MPI

- To compile MPI
  - Run: /usr/lib64/openmpi/1.4-gcc/bin/mpicc (or mpic++)
    - Pro Tip: Add the line: export PATH=/usr/lib64/openmpi/1.4-gcc/bin:$PATH to your .bashrc end your submission script and you don't need the full path
  - Replace gcc/g++ in your favorite makefile with this call

Compiling MPI

- The make command is your friend
  - If you're on a compute node "make –j4" will compile your code using 4 threads
    - Saves you time
    - Substitute the number of slots you have for 4
  - Combine qrsh (see earlier) with this to save yourself some time

Running MPI Jobs

```
#!/bin/bash
#$ -cwd
#$ -pe orte 4
#$ -N orte_job
#$ -M jhegie
#$ -m bes
#$ -q ECSL.owner

export PATH=/usr/lib64/openmpi/1.4-gcc/bin:$PATH

mpirun –np $NSLOTS ./mpijob
```

Running MPI Jobs

- Most of that should be familiar
- But what is this “orte” thing?
  - orte is the name of the standard MPI environment
  - It uses all available slots on a machine before moving to the next one
- The export lets you not use fully qualified paths
  - SGE doesn’t actually parse your bashrc when it spawns your job
- mpirun starts your job
  - -np tells it now many MPI processes will be present
  - $NSLOTS is an environment variable set by SGE
    - 4 in the example
  - The last argument is the program to run

Parallel Environments

- SharedMem
  - Used to allocate shared memory jobs on one, and only one, machine
- orte
  - Fills up all available slots on a machine before spilling over to the next one
- orte-rr
  - Like orte, but fills slots round robin (one per machine until all available machines have a slot taken