Programming Environment
on Ranger Cluster

Cornell Center for Advanced Computing
December 8, 2010
User Guides

• TACC
  – Ranger (http://services.tacc.utexas.edu/index.php/ranger-user-guide)
  – Spur (http://services.tacc.utexas.edu/index.php/spur-user-guide)

• CAC

• Tutorials
  – Beginners Unix (http://info.ee.surrey.ac.uk/Teaching/Unix/)
Just Like Your Computer

SSH to Head Node

Scheduler

Head Node
Password Change

Change your TACC password (for your workshop account)

https://tas.tacc.utexas.edu/TASMigration/ChangeTASPassword.aspx

Change your TeraGrid password (if you get a TeraGrid account)

https://www.teragrid.org/web/user-support/passwords
SSH Clients

- Windows: Putty
- Linux: built-in as “ssh”
- Mac: built-in as “ssh”

Login now to ranger.tacc.utexas.edu:

```
ssh train1xx@ranger.tacc.utexas.edu
-or-
putty
```

Host Name: ranger.tacc.utexas.edu
Account Info

- Find your account number at bottom of splash screen.

<table>
<thead>
<tr>
<th>Name</th>
<th>Avail SUs</th>
<th>Expires</th>
</tr>
</thead>
<tbody>
<tr>
<td>20101208HPC</td>
<td>5000</td>
<td>2010-12-16</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Disk</th>
<th>Usage (GB)</th>
<th>Limit</th>
<th>%Used</th>
<th>File Usage</th>
<th>Limit</th>
<th>%Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>/share</td>
<td>0.0</td>
<td>6</td>
<td>0.00</td>
<td>47</td>
<td>100000</td>
<td>0.05</td>
</tr>
<tr>
<td>/work</td>
<td>0.0</td>
<td>350</td>
<td>0.00</td>
<td>1</td>
<td>2000000</td>
<td>0.00</td>
</tr>
</tbody>
</table>
Experiment

echo $SHELL
chsh –l
man chsh
env  (show environment variables – persists)
set  (show shell variables – current shell only)
pwd
ls –la
cat .login
cat /usr/local/etc/login
cat .login_user  (create then edit this one to personalize)
Un-TAR Lab Files

- TAR = Tape archive.
- Just concatenates files.
- `tar <switches> <files>`
  - `z` = compress or decompress
  - `x` = extract
  - `c` = create
  - `v` = verbose
  - `t` = list files
  - `f` = next argument is the file to read or write
- `~userid` is the home directory of that user
- For example, to create a tar: `tar cvf myfiles.tar dir1 dir2 README`

Get the lab files:

```
tar zxvf ~train103/envi.tgz
```
Software

Software section in User Guide

Software list available on Ranger

The module utility is used to provide a consistent, uniform method to access software.
MODULE Command (Ranger-only)

- Affects $PATH, $MANPATH, $LIBPATH
- Load specific versions of libraries/executables
- Works in your batch file
- Define environment variables:
  - TACC_MKL_LIB, TACC_MKL_INC, TACC_GOTOBLAS_LIB
Try MODULE

```
module
module list
module help < module_name >
module load < module_name >
module avail
module load intel  # note the response
module swap pgi intel  # so delete pgi
module load fftw2
module del fftw2
```

Order matters! Unload MPI, then choose a compiler, then load the MPI version.
MODULE Examples

login4% module list
Currently Loaded Modules:

1) TACC-paths
2) Linux
3) cluster-paths
4) pgi/7.2-5
5) mvapich/1.0.1
6) binutils-amd/070220
7) TERAGRID-paths
8) gx-map/0.5.3.3
9) srb-client/3.4.1
10) tg-policy/0.2
11) tgproxy/0.9.1
12) tgresid/2.3.4
13) tgusage/3.0
14) uberftp/2.6
15) tginfo/1.0.1
16) TERAGRID-BASIC
17) globus/4.0.8
18) GLOBUS-4.0
19) TERAGRID-DEV
20) CTSSV4
21) gzip/1.3.12
22) tar/1.22
23) cluster
24) TACC

login4% module avail
----------------------------------- /opt/apps/pgi7_2/modulefiles -----------------------------------

acml/4.1.0
autodock/4.0.1
fftw3/3.1.2
glpk/4.40
gotoblas/1.26 (default)
gotoblas/1.30
gotoblas2/1.00
gotoblas2/1.05 (default)
hdf5/1.6.5
hecura-debug/1.5rc2
hecura/1.5.1
metis/4.0
mvapich-old/1.0.1
mvapich/1.0.1
mvapich2-debug/1.2
mvapich2-new/1.2
mvapich2/1.2
nco/3.9.5
netcdf/3.6.2
openmpi/1.3
Now Swap Compilers

Swap compilers and look again:

login4% module swap pgi intel
login4% module avail

------------------------------------------ /opt/apps/intel10_1/modulefiles ------------------------------------------

acml/4.1.0    hecura-debug/1.5rc2    mvapich/1.0
autodock/4.0.1 hecura/0.1         mvapich/1.0.1 (default)
boost/1.37.0   hecura/1.4         mvapich2-new/1.2
boost/1.39.0   hecura/1.4rc2     mvapich2/1.2
boost/1.41.0 (default) hecura/1.5 (default) nco/3.9.5
fftw3/3.1.2    hecura/1.5rc2    netcdf/3.6.2
glpk/4.40      hecura/trunk_2009_09_20 octave/3.2.4
gotoblas/1.26 (default) hmmmer/2.3.2 openmpi/1.2.4
gotoblas/1.30   metis/4.0       openmpi/1.2.6
gotoblas2/1.00 mvapich-devel/1.0 openmpi/1.3 (default)
gotoblas2/1.05 (default) mvapich-old/1.0.1
hdf5/1.6.5      mvapich-ud/1.0

12/8/2010
Two Time Commands

- Use `time` or `/usr/bin/time` to see how long your program runs and estimate if it’s having gross difficulties
- `/usr/bin/time` generally gives more information

```
make
time ./hello
/usr/bin/time ./hello
```
Submit a Job

1. Write a script

```
#!/bin/sh
echo Starting job
date
/usr/bin/time ./hello
date
echo Ending job
```

2. Add batch instructions

```
#!/bin/sh
#$ -N hello
#$ -cwd
#$ -o $JOB_NAME.$JOB_ID
#$ -j y
#$ -q development
#$ -pe 1way 16
#$ -V
#$ -l h_rt=00:2:00
echo Starting job
date
/usr/bin/time ./hello
date
echo Ending job
```

3. Submit it to the scheduler

```
qu -A 20101208HPC job.sge
```
How Are the Queues?

qconf –sql  # List available queues
qconf –sq <queue name>  # Soft and hard wall clock limits

cat /share/sge6.2/default/tacc/sge_esub_control  # Queue core limit
showq
showq –u
qdel or qdel –f  # Delete job
Queue Examples

```
login3% qconf -sql
clean
development
large
long
normal
request
reservation
serial
sysdebug
systest
vis
```

```
login3% qconf -sq development
qname development
qtype BATCH INTERACTIVE
pe_list 16way 15way 14way 12way 8way 4way 2way 1way
slots 16
s(tmpdir /tmp
shell /bin/csh
prolog /share/sge/default/pe_scripts/prologWrapper
epilog /share/sge/default/pe_scripts/tacc_epilog
shell_start_mode unix_behavior
s_rt 07:58:00
h_rt 08:00:00
```

Slots = number of cores, 16 per node
pe = wayness, how many cores per node
Job is killed if over time limit.

Why 15way?
States

- Unscheduled – Likely not good
- DepWait – You can ask that one job run after another finishes.
- w(aiting) – Queued, waiting for resources to run.
- r(unning) – As far as SGE is concerned, it’s going.
- h(old)
- s(uspending)
- E(rrror)
- d(letion)
Submit a Job Example

cat makefile
make
ls –la
./hello
less job.sge
qsub –A 20101208HPC job.sge
showq –u
less hello.oNNN
env | sort > z.txt
diff z.txt hello.oXXX | less

# Review the makefile
# Compile hello.c
# Take a look at what compiled
# Run compiled program
# View the script
# Submit the job
# Check job status
# Look at the output file
# Save login env vars to file
# Compare login vs batch env vars
Environment Variables in Batch

> ENVIRONMENT=BATCH
> HOSTNAME=i182-401.ranger.tacc.utexas.edu
> JOB_ID=743637
> JOB_NAME=hello
> JOB_SCRIPT=/share/sge6.2/execd_spool//i115-306/job_scripts/1715044
> NHOSTS=1
> NQUEUES=1
> NSLOTS=16
> OMP_NUM_THREADS=1
> PE=1way
> PE_HOSTFILE=/share/sge6.2/execd_spool//i115-306/active_jobs/1715044.1/pe_hostfile
> QUEUE=development
> SGE_ACCOUNT=20101208HPC
> SGE_O_SHELL=/bin/tcsh
> SGE_O_WORKDIR=/share/home/0000/train135
> SGE_ROOT=/opt/sge6.2
> SGE_RSH_COMMAND=/usr/bin/ssh -q
> SGE_STDERR_PATH=/share/home/0000/train135/hello.o1715044
> SGE_STDIN_PATH=/dev/null
> SGE_STDOUT_PATH=/share/home/0000/train135/hello.o1715044
Parallel Environment

- Each node has 16 cores and is used by one person at a time

- `#$ -pe 1way 16` Run one task on a node with 16 cores
- `#$ -q serial`
- `./hello`

- `#$ -pe 8way 64` Run 8 tasks/node on 4 nodes
- `#$ -q normal`
- `export MY_NSLOTS=31` Launch 31 tasks
- `lbrun ./a.out` Run with mpi wrapper
To Edit A File in VI (short for “visual”)

• “vi filename” will open it or create it if it doesn’t exist.
• Command mode and Insert mode. You start in command mode.
• Command mode. Cursors work here, too.
  – :w  Writes a file to disk.
  – :q  Quits
  – :q! Quits even if there are changes to a file
  – i  Takes you to insert mode
• Insert Mode
  – Cursors, typing characters, and deleting work here.
  – Escape key takes you to command mode.
• Ctrl-c will get you nowhere.
X-Windows

- Start X-Windows server on local machine.

```
$ echo $DISPLAY
localhost:39.0
$ emacs README
```

```
$ jobs
$ kill %1
```
Login with X-Windows

- Start Exceed->Exceed on Windows Startup menu (Already started on Mac and Linux)
- ssh –X on Linux, Mac. For Windows, select in Putty Connection->SSH->X11, and check “X11 Forwarding”
- Type in username and password.
- echo $DISPLAY
- emacs README&   # This runs emacs in the background.
- At the command prompt, type “jobs” to see that you have a backgrounded job.
- Try Emacs for a while, then kill it with
- kill %1
VNC

• **VNCServer**
  – used to start a VNC (Virtual Network Computing) desktop.
  – a Perl script which simplifies the process of starting an Xvnc server.
  – can be run with no options at all. In this case it will choose the first available display number.
• VNCServer copies a bitmap of the X-Windows screen across.
• Can be much less chatty than X-Windows.
• Good for remote graphics.
• VNCServer screen 4 uses TCP/IP port 5904.
Connect with VNC

• Start VNC on Ranger
  – First ssh normally.
  – Type “vncserver” and look for screen number, for example. “4”.

• Connect with a client
  – RealVNC or TightVNC on Windows
  – On Linux, vinagre or vncviewer
  – Connect to “ranger.tacc.utexas.edu:4” or your port number

• Be sure to kill it when you are done
  – vncserver –kill 4
login3% vncserver

New 'login3.ranger.tacc.utexas.edu:1 (train200)' desktop is
login3.ranger.tacc.utexas.edu:1

Starting applications specified in /share/home/0002/train200/.vnc/xstartup
Log file is /share/home/0002/train200/.vnc/login3.ranger.tacc.utexas.edu:1.log

login3% vncserver -kill :1
Killing Xvnc process ID 11406
Questions?