Using Intersect's partnershare of Raijin

Dr. Joachim Mai

Aug2013

Getting Resources

- Resource allocation round in Oct for the coming year.
- Smaller allocations (<20 kSUs per quarter) possible any time
- Use the forms:

http://nf.nci.org.au/accounts/forms/user_registration.php

https://nf.nci.org.au/accounts/projects_new/APP_form.php

- Connect to existing project:

http://nf.nci.org.au/accounts/forms/user_connection.php

INTERSECT

Account Details

- Once the CI approves your connection to the project your account is set up and you are sent an email with account

details.

- User names are of the form abc123 - abc for your initials and 123 for partner.
- Passwords are sent by SMS to the mobile number provided when you registered.
- Passwords can be given over the phone if necessary, but not by email.
- Use the passwd command to change this when you first log in.

INTERSECT

Raijin (July 2013)

- Fujitsu Primergy
- Distributed memory machine
- 57,472 Sandy Bridge cores
- 158 TB RAM
- 9 PB disk
- Infiniband FDR
- Centos 6.4
- Intersect owns about 4%

Remark: Orange (Dec 2012)

- SGI Cluster
- Distributed memory
- 1,600 Sandy Bridge cores
- 10 x 256GB RAM
- 90 x 64 GB RAM
- 100 TB disk
- Infiniband QDR
- SLES 11

INTERSECT

Differences Orange ↔ Raijin

- Memory:

  Orange: 90 x 64 GB

  10 x 256 GB

  Raijin: 2395 x 32 GB

  1125 x 64 GB

  72 x 128 GB

- Software
- Size: 100 nodes (Orange) vs 3,592 nodes (Raijin)

INTERSECT

Raijin Stakeholders

- NCMAS 15%
- CSIRO 21.4%
- BOM 18.9%
- ANU 17.7%
- Flagships 5.0%
- INTERSECT 3.8%
- GA 3.4%
- Monash 1.7%
- UNSW 1.7%
- UQ 1.7%
- USyd 1.7%
- University of Adelaide 1.7%
- Director's share and others 4%

INTERSECT

Disks Raijin

- Lustre parallel global FS
- 10 PB
- 300GB scratch disks per node

INTERSECT

Project Accounting

- Each user belongs to one or more projects
- To change or set the default project, edit your .rashrc file in your home directory, and change the PROJECT variable as desired. A typical .rashrc file looks like

```
setenv PROJECT c25
setenv SHELL /bin/bash
```

- Display usage:

```
nci_account -P project -p 2013.q3 -v
```

INTERSECT

Connecting to Raijin

- ssh -Y username@raijin.nci.org.au
- For connecting under Windows use putty
- For file transfer use scp, sftp or a GUI client such as Filezilla

Setting the Environment

module list To see the modules loaded

module avail To see available modules

module show To see the commands that are carried out in the module

module load To load the environment settings package

module unload To remove a previously loaded software package. This is useful in situations where different package settings clash (multiple MPIs for example)

Exercise 1

- Log into Raijin. Your project code is c25 and you username is aaa777.
- Read the message of the day (MOTD).
- Try the following commands:

```
hostname          # to see the node you are logged into
nci_account       # to see the current state of the project
printenv          # to look at your environment settings
module list       # to check which modules are loaded on login
module avail      # to see which software packages are installed
module show pbs   # to see what environments are set by a module
```

INTERSECT

Getting information and help

- Message of the Day - MOTD

- Downtime http://nf.nci.org.au/notices_news/

- Userguide at http://nf.nci.org.au/wiki/RaijinUserGuide

- FAQs at http://nf.nci.org.au/facilities/faq

- Software pages at http://nf.nci.org.au/facilities/software

-Email to hpc_support@intersect.org.au or help@nf.nci.org.au for NCI related problems

INTERSECT

What filesystem to use

Source code and important input files: /home

Job input/output files: /short

Temporary or scratch files: JOBFS

Long term archived files: MDSS

Processing of large data files: /projects

Filesystem overview

/home: for program, 2GB default, backup, global availability, permanent

/short: for I/O, 80GB, no backup, global availability

/JOBFS: for heavy I/O100MB, no backup, local to node, during job

MDSS: for archiving, 20GB, 2 copies, external accessing, permanent

Monitoring disk usage

```
$ lquota
---------------------------------------------------------------
fs Usage Quota Limit iUsage iQuota iLimit
---------------------------------------------------------------
mhk900 home 20.7MB 0kB 0kB 421 0 0
z00 short 1153GB 0kB 1500GB 1527750 0 10000000
z10 short 0kB 0kB 78.0GB 1 0 200000
c25 short 428kB 0kB 200GB 107 0 200000
y03 short 557GB 0kB 10.0TB 163891 0 1000000
z29 short 0kB 0kB 78.0GB 0 0 200000
ua6 short 0kB 0kB 195TB 0 0 200000
c25.data short 0kB 0kB 0kB 0 0 0
z34 short 636kB 0kB 78.0GB 29 0 200000
---------------------------------------------------------------
```

INTERSECT

Optimise usage

Lots of small IO to /short (or /home) can be very slow and can severely impact other jobs on the system.
Avoid "dribbly" IO, e.g. writing 2 numbers from your inner loop.
Writing to /short every second is far too often!

Avoid frequent opening and closing of files (or other file operations)

Use /jobfs instead of /short for jobs that do lots of file manipulation

To achieve good IO performance, try to read or write binary files in large chunks (of around 1MB or greater)

Exercise 2

- Use the commands lquota and du to determine the disk space available to you in /home and /short.
- Have a look at your /short. Any user of the project can write data here.

```
cd /short/$PROJECT
ls -ld .
ls -l DATA
ls $USER
```

Extract the examples here:

```
 cd $USER
tar xvf /short/c25/intro_exercises.tar
cd INTRO_COURSE
pwd
ls -l
```

Exercise 2 (cont)

- Change the permissions on your files and directories to allow/disallow others in your group to access them.

```
man chmod
chmod g+r filename   # allow group read to filename
chmod g-r filename   # disallow group read to filename
chmod g+w filename # allow group write to filename
chmod g+x filename  # allow group execute to filename
```

Exercise 2 (cont)

- Use the MDSS with the following commands:

```
 cd /short/$PROJECT/$USER
mdss get Data/data.tar
ls -l
tar xvf data.tar
ls
rm data.tar
mdss mkdir $USER
netmv -t $USER.tar DATA $USER
nqstat
more DATA.o*
mdss ls $USER
mdss rm $USER/$USER.tar
```

Compiling and Optimising

- We recommend using the Intel compilers (icc, ifort, icpc).
- Check which versions are available:

module avail intel-fc

module avail intel-cc

- Read the manual pages (man icc, man ifort, man icpc)
- Compiling and linking of a Fortran code:

ifort -o matmulf matmul.f

- Options: default is O2 for Intel and O0 for Gnu compilers
- O0 means no optimisation and is very slow (-g implies -O0)
- O3 means aggressive optimisation, be careful.

Exercise 3

- Compile the sample code matmulf.f and matmulf.c with the Intel compilers using O3 and O0 optimisation. Compare the runtime using the time command.

- Compile the code netcdfex.f using the netcdf library:

```
module load netcdf
module show netcdf
module list
cat netcdfex.f
ifort -o netcdfex netcdfex.f -lnetcdff -lnetcdf
netcdfex
ncdump simple_xy.nc
```

The batch system PBSPro

- The batch system distributes work evenly over the system and ensures that jobs cannot impact each other (e.g. exhaust memory or other resources)

Raijin uses a customised version of PBSPro on Raijin. It is currently being tuned.

- Batch queues:
normal: default
express: 3 times the charges of Sus
copy: to copy data, e.g. to MDSS

Using PBS

- Submit your job via qsub
- Specify resources (walltime, mem, nodes PBS_JOBFS etc).
- Read the handbook!!!

Scheduling
- Jobs start when sufficient resources get available
- Jobs can be suspended when jobs with higher priority wait
- Priority depends on the amount of resources available

Use check pointing for longer running jobs

Monitoring jobs

Useful commands:

```
qstat          # show the status of the PBS queues
nqstat         # enhanced display of the status of the PBS queues
qstat -s       # display additional comment on the status of the job
qps jobid      # show the processes of a running job
qls jobid      # list the files in a job's jobfs directory (to come)
qcat jobid     # show a running job's stdout, stderr or script
qcp jobid      # copy a file from a running job's jobfs directory (to come)
qdel jobid     # kill a running job
```

Exercise 4

Submit the following PBS script via qsub:

```
#!/bin/csh
#PBS -l wd
#PBS -q express
#PBS -l walltime=00:10:00,mem=52MB
#PBS -P c25
time ./matmuls
time ./matmulf
```

And monitor it's prgress via:

```
nqstat
qps
jobid
pbs_rusage
jobid
```

Exercise 4 (cont)

Try an interactive batch job:

```
[aaa777@raijin5 ~]$ qsub -I -l walltime=00:10:00,mem=500Mb,wd -P c25
qsub: waiting for job 215984.r-man2 to start
qsub: job 215984.r-man2 ready
[aaa777@r73 ~]$ hostname
r73
[aaa777@r73 ~]$ module list
Currently Loaded Modulefiles:
1) pbs 3) intel-cc/12.1.9.293
2) intel-fc/12.1.9.293 4) openmpi/1.6.3
[aaa777@r73 ~]$
[aaa777@r73 ~]$
qsub: job 215984.r-man2 completed
```

Exercise 5

Compile and run the following OpenMP code

```
 ifort -O3 -openmp matmul_omp.f -o matmul_omp
export OMP_NUM_THREADS=2
time matmul_omp
export OMP_NUM_THREADS=4
time matmul_omp
```

Exercise 5 (cont)

Compile and run the following MPI code
module list
mpif90 mpiexample1.f -o mpiexample.exe
mpirun -np 4 mpiexample.exe
or for a more complicated example:
mpif90 mpiexample2.f -o mpiexample.exe
mpirun -n 4 mpiexample.exe
mpirun is the usual instruction to start an MPI program.
man mpirun
for further details on usage.
C code simple example:
mpicc mpiexample3.c -o mpiexample.exe
mpirun -np 4 mpiexample.exe
and for a more complicated code:
mpicc mpiexample4.c -o mpiexample.exe
mpirun -n 4 mpiexample.exe

Profiling

On a system level use: top, iostat, vmstat
On PBS level use: qstat, qstat -f, qps etc.

Lightweight: IPM
Heavyweight: Vampir

General profiling:

```
$ ifort -p -o prog.exe jacobi_serial.f
$ ./prog.exe < input.1
$ gprof ./prog.exe gmon.out
```

For the GNU compilers do

```
$ gfortran -pg -o prog.exe jacobi_serial.f
$ ./prog.exe < input.1
$ gprof ./prog.exe gmon.out
```

Example:use of PBS_JOBFS

```
#!/bin/bash
#PBS -q express
#PBS -l walltime=2:00
#PBS -l jobfs=10mb
#PBS -l mem=30mb
cd $PBS_JOBFS
echo "Moving files from short directory to the local directory"
cp /short/$PROJECT/$USER/input.1 .
cp /short/$PROJECT/$USER/jacobs .
# Run program and write an output file to the local disk.
time ./jacobs <input.1 > output$PBS_JOBID 2>&1
# Move output data to /short space.
echo "The output files are now on my /short space."
mv output$PBS_JOBID /short/$PROJECT/$USER
# Archive to MDSS using netcp
cd /short/$PROJECT/$USER
netcp -N save_data output$PBS_JOBID $USER/output$PBS_JOBID
```

INTERSECT

Acknowledgement

This course is based on a course created by Margaret Kahn/NCI and presented to the BOM. Margaret and the NCI staff is greatly acknowledged for help, support and discussions.

INTERSECT

Thanks for attending!

Please complete our course survey at:

http://svy.mk/18c8dHa

Any further questions, please contact us at

training@intersect.org.au

INTERSECT