ARSC

Introduction to Pacman

David Newman

(From slides by Tom Logan)

Overview

- **Hardware**
- **Programming Environment**
- **Compilers**
- **Queueing System**

PACMAN –
Penguin Computing Opteron Cluster

- **12 Login Nodes:**
  - 2- Six core 2.2 GHz AMD Opteron Processors; 64 GB memory per node
- **256 Four Core Compute Nodes**
  - 2- Dual core 2.6 GHz AMD Opteron Processors; 16 GB memory per node
- **88 Sixteen Core Compute Nodes**
  - 2 Eight core 2.3 GHz AMD Opteron Processors; 64 GB memory per node
  - QLogic QDR Infiniband; 250 GB local disk
- **20 Twelve Core Compute Nodes**
  - 2 Six core 2.2 GHz AMD Opteron Processors; 32 GB memory per node
- **4 Large Memory Nodes**
  - 4 Eight core 2.3 GHz AMD Opteron Processors; 256 GB memory per node
  - QLogic QDR Infiniband; 1 TB local disk; 80 GB solid state drive
- **2 GPU Nodes**
  - 2 Quad core 2.4 GHz Intel CPUs; 64 GB of memory per node

Login Node Usage

Login Node	Intended Purpose
pacman1.arsc.edu pacman2.arsc.edu	Compiling and Batch Job Submission
pacman3.arsc.edu through pacman9.arsc.edu	Compute Intensive Interactive Work
pacman10.arsc.edu through pacman12.arsc.edu	Batch Data Transfer Work
pacman13.arsc.edu	Compute Intensive Interactive Work / 256GB Memory / 32 Cores

Pacman Storage

- **Pacman supports the ARSC standard storage locations**
  - $HOME: small, backed up, not purged
  - $CENTER: large, not backed up, purged
  - $ARCHIVE: no quota, backed up, not purged.
  - NOTE:
    - $ARCHIVE not available from compute nodes! $ARCHIVE is only accessible to the pacman login nodes
    - ARSC recommends that you avoid accessing $HOME in parallel jobs

Parallel Programming Models

Level	Model	Description
Node	Auto	Automatic parallelization of basic loops. Only available with PGI compilers (use –Mconcur= option)
Node	OpenMP	Explicit shared memory model using directives to achieve loop level parallelism (use –mp option with PGI compilers)
System	MPI	Most common and portable method for scalable distributed memory parallelism

Modules

- **Modules are fully supported on Pacman for many common packages including compilers:**

Module Name	Description
PrgEnv-pgi	Programming environment using the PGI compilers & OpenMPI stack (default module).
PrgEnv-gnu	Programming environment using GNU compilers & OpenMPI stack.

Sample Module Commands

Command	Example Use	Purpose
module avail	module avail	lists all available modules for the system.
module load <*pkg*>	module load PrgEnv	loads a module file from the environment
module unload <*pkg*>	module unload PrgEnv	unloads a module file from the environment
module list	module list	displays the modules which are currently loaded.
module switch old new	module switch PrgEnv-pgi PrgEnv-gnu	replaces the module old with module new in the environment
module purge	module purge	unload all module settings, restoring the environment to the state before any modules were loaded.

Module Use

- **Modules make your life easier by setting up your environment for various packages!**
- **Currently available modules include:**
  - OpenFOAM, abaqus, cmake, cuda, ferret, git, grads, idl, jdk, matlab, nco, ncl, ncview, nwchem, octave, paraview, petsc, python, r, subversion, tau, totalview, …
- **More continue to be added as the system matures**

Compilers on Pacman

Item	MPI	PGI	GNU
Fortran 77	mpif77	pgf77	g77
Fortran 90/95	mpif90	pgf90	gfortran
C	mpicc	pgcc	gcc / cc
C++	mpiCC	pgCC	g++ / c++
Debugger		pgdbg	gdb
Performance		pgprof	gprof
Default Module	PrgEnv-pgi PrgEnv-gnu	PrgEnv-pgi	PrgEnv-gnu

Compiling

Fortran

```
gfortran -o xanal_rs anal_rs.f
```

or C++

```
g++ -o bin/standard \
      src/main.cpp \
      src/bhat.cpp \
      src/simplex.cpp \
      src/ivector.cpp \
      src/flows.cpp \
```

A few PGI compiler options

-c	Generate object file but don't link
-g	Add debugging information
-O3	Higher level of optimization (default is –O2)
-fast	Higher level of optimization than –O3
-Mipa	Perform interprocedural analysis
-Minfo	Report optimizations that are made
-Mneginfo	Report optimizations that are not made
-Mconcur	Enable autoparallelization
-mp	Enables parallelization via OpenMP directives

Batch System

- **Batch queueing**
  - Allows job scheduling on shared compute nodes
  - Requires users to specify resource requests
  - Ensures that nodes are shared fairly
  - Manages resources to maximize throughput
- **Pacman uses the Torque/Moab batch queueing system, which is based on PBS, the Portable Batch System**

Batch Script Contents

1. Queueing Commands

- Shell dialect (e.g. bash, ksh)
- Execution queue to use
- Job walltime
- Number of nodes required

2. Commands to Execute

- File/Directory Manipulations
- Code to execute

Torque Script - Big mem Example

```
#!/bin/bash
#PBS -q bigmem
#PBS -l walltime=220:00:00
#PBS -l nodes=1:ppn=1
#PBS -j oe
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/u1/uaf/newman/CLP/
Clp-1.14.8/lib/
cd $PBS_O_WORKDIR

./clp-opt-O2-primal
```

Torque Script - Shared Example

```
#!/bin/bash
#PBS -q shared
#PBS -l walltime=20:00:00
#PBS -l nodes=1:ppn=1
#PBS -j oe
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/u1/uaf/newman/
CLP/Clp-1.14.8/lib/
cd $PBS_O_WORKDIR

./clp-opt-O2-primal
```

Torque Script - MPI Example

```
#!/bin/bash
#PBS -q standard
#PBS -l walltime=8:00:00
#PBS -l nodes=4:ppn=12
#PBS -j oe

cd $PBS_O_WORKDIR

mpirun ./myprog
```

Torque Script - MPI Example

```
#!/bin/bash
#PBS -q standard
#PBS -l walltime=8:00:00
#PBS -l nodes=6:ppn=12
#PBS -j oe

cd $PBS_O_WORKDIR

# Use only 2 PEs per node
mpirun -np 12 -npernode 2 ./myprog
```

Torque Script - OpenMP

```
#!/bin/bash
#PBS -q standard
#PBS -l walltime=8:00:00
#PBS -l nodes=1:ppn=12
#PBS -j oe

cd $PBS_O_WORKDIR
export OMP_NUM_THREADS=12

./myprog
```

Torque Script – Shared Node

```
#!/bin/bash
#PBS -q shared
#PBS -l walltime=8:00:00
#PBS -l nodes=1:ppn=1
#PBS -j oe

cd $PBS_O_WORKDIR

./myprog
```

Torque Script - Data Staging

```
#!/bin/bash
#PBS -q transfer
#PBS -l walltime=8:00:00
#PBS -l nodes=1:ppn=1
#PBS -j oe

cd $PBS_O_WORKDIR
batch_stage $ARCHIVE/mydataset/*
cp -r $ARCHIVE/mydataset/* . || exit 1

qsub mpi_job.pbs
```

Common PBS commands

- `qsub job.pbs`- **queue the script "job.pbs" to be run by PBS.**
- `qstat`- **list jobs which haven't yet completed**
- `qdel` *`jobid`*- **delete a job with ID=jobid from the queue.**
- `qmap`- **show a graphical list of the current work on nodes**

Common Queues

- `standard`- **regular work. This queue requires that you have an allocation of CPU time.**
- `debug`- **for quick turn around debugging work.**
- `background`- **lowest priority queue, but doesn't require that you have an allocation.**
- `shared` – **shared node jobs, for running serial codes on shared nodes**
- `transfer`- **For data transfer to and from $ARCHIVE. Be sure to bring all $ARCHIVE files online using batch_stage prior to the file copy.**
- **Use "news queues" to find out more details on number of nodes and walltime allowed per queue**

Additional Information

- **Watch www.arsc.edu for upcoming training and additional information about ARSC**
- **More info at**
  - ARSC Howto: http://www.arsc.edu/acad/support/howtos/usingpacman.html
  - PGI Compilers: http://www.pgroup.com/resources/docs.htm
  - GNU Compilers: http://gcc.gnu.org/
- **ARSC Help Desk**

  Phone: (907) 450-8602

  Email: consult@arsc.edu

  Web: http://www.arsc.edu/support