Description of the Euler Cluster

Andrew Seidl

Wisconsin Applied Computing Center

26 July 2013

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

Outline

1. Hardware

2. Software

3. Using TORQUE

Funding

- Grants from US Army Research Office
- Various research groups from across campus
- Hardware donations from:
    - AMD
    - NVIDIA

Hardware Overview

- 86.7 Tflops SP (GPU)
- 11.5 Tflops DP (GPU)
    - 10.0 Tflops DP (CPU)
- 60 NVIDIA GPUs
- 1 Intel Xeon Phi
- 1148 CPU cores
    - 1024 AMD
    - 124 Intel
- 2.7 TB RAM
- 36 TB storage
- 40Gbps Infiniband interconnect (QDR)

NVIDIA GPU Compute Nodes

- 40x GeForce GTX 480
- 8x GeForce GTX 680
- 8x Tesla C20x0
- 4x Tesla K20x
- 14x main GPU compute nodes:
  - 2x Intel Xeon E5520 2.26GHz
  - 48GB RAM
  - 4x GPUs
- 1x dev GPU node:
  - 2x Intel Xeon E5-2630 2.30GHz
  - 64GB RAM
  - 4x Tesla K20x
    - sometimes 3x, plus Xeon Phi
- Torque resource flags: gpu, tesla, kepler

AMD CPU Compute Nodes

- 16x CPU compute nodes
  - 4x AMD Opteron 6274 2.20GHz
    - total of 64 cores per machine
  - 128GB RAM
- Torque resource flags: cpu, amd

File server

- Two identical systems, mirrored
- 24x 2TB WD RE4 drives
- RAID6
- NO BACKUPS
  - be sure to push your code to another server

Software Overview

- Scientific Linux 6.2
    - upgrade to CentOS 6.4 planned
- xCAT for node management
- TORQUE for job scheduling and management
    - might switch to HTCondor, pending discussions next week

Available Software

- Development
  - AMD CodeAnalyst
  - Boost 1.5{1,2,3}
  - Cray Chapel
  - CUDA {3.2,4.0-4.2,5.0,5.5}
  - GCC 4.{4,6,7,8,9}
  - Intel Compilers (license req'd)
  - Python {2.6,2.7,3.3}
- MPI
  - Intel MPI (license req'd)
  - MVAPICH
  - MVAPICH2
  - OpenMPI

Other Software

- Graphics
  - Blender
  - LuxRender
  - Pixar Renderman
  - Paraview
  - Point Cloud Library (PCL)
  - VTK
- Engineering
  - MATLAB (license req'd)
  - Mathematica (license req'd)
  - MSC ADAMS (license req'd)

Job Configuration

- Jobs are configured and run via basic shell scripts (˜/Example Jobs/gpu-scan.sh):

```
#!/bin/sh
#PBS -N gpu-scan
#PBS -l nodes=1:gpus=1,walltime=00:01:00

cd $PBS_O_WORKDIR
$NVSDKCOMPUTE_ROOT/C/bin/linux/release/scan
```

`-N gpu-scan`: job name is `gpu-scan`

`nodes=1`: give me one node, one CPU

`gpus=1`: give me one GPU

`walltime=00:01:00`: this job only requires one minute of wall time

Submitting Jobs

- To submit:
  `qsub ~/Example Jobs/gpu-scan.sh`
- For an 'interactive job', use `qsub -I`
  - if you need X11, use `qsub -X -I`
    - be sure you forwarded X for you SSH session: `ssh -X euler`
  - you can use `-l` resource flags for different resources:
    `qsub -I -l nodes=1:ppn=8:gpus=2:tesla`

Monitoring Jobs

- Use `qstat` to monitor job status
  `qstat -u $USER`
  - `S` column shows status: `Q`: queued, `R`: running, `C`: complete
- On completion, TORQUE will place `stdout`, `stderr` messages under jobname-{o,e}jobid