UF Research Computing GPU Resources and GPU-based Jobs

Charles Taylor
UF HPC Center
April 22, 2013

UF Research Computing

UNIVERSITY of FLORIDA

- Mission
  - Improve opportunities for research and scholarship
  - Improve competitiveness in securing external funding
  - Provide high-performance computing resources **and support** to UF researchers

UF Research Computing

- GPUs
  - Graphics Processing Unit
  - Highly Parallel Hardware (hundreds of "cores")
  - Single Precision Floating Point
  - Double Precision Floating Point
  - Local Device Memory Hierarchy
  - Originally designed for highly parallel vertex transformation and image (texture) mapping
  - General purpose parallel processing for suitable applications

UF Research Computing

- Available GPU Resources
  - 8 S1070 (Tesla)
    - 240 Thread Processors
    - 4 GB Device RAM
  - 16 M2070 (Fermi)
    - 448 Thread Processors
    - 6 GB Device RAM
  - 72 M2090 (Fermi)
    - 512 Thread Processors
    - 6 GB Device RAM

UF Research Computing

- Policies
  - Some GPUs available for general use
  - Most dedicated to research projects
  - Accessible via the Batch System
  - Must have access to "gpu" queue
  - Investors have priority

UF Research Computing

- Interactive Use
  - `qsub -I -l nodes=1:gpus=1:tesla,walltime=01:00:00 -q gpu`
  - `qsub -I -l nodes=1:gpus=1:fermi,walltime=01:00:00 -q gpu`
  - `qsub -I -l nodes=1:gpus=2,walltime=01:00:00 -q gpu`
  - `qsub -I -l nodes=tesla1:gpus=1,walltime=01:00:00 -q gpu`
- http://wiki.hpc.ufl.edu/doc/NVIDIA_GPUs

UF Research Computing

- Must Use the Assigned GPU
  - Some applications use "--device" argument
  - CUDA Runtime supports CUDA_VISIBLE_DEVICES environment variable
  - Export CUDA_VISIBLE_DEVICES=2
  - Restricts the application to using gpu 2

UF Research Computing

- Must Use the Assigned GPU
  - In a program…

```
// Choose a cuda device for kernel execution
cudaRet = cudaSetDevice(gpuDev);
if (cudaRet == cudaErrorInvalidDevice) {
   perror("cudaSetDevice returned Invalid Device");
}
else {
// Was device selected?
cudaGetDevice(&getDev);
printf("cudaGetDevice()=%d\n",getDev);
}
```


UF Research Computing

- Must Use the Assigned GPU
  - In a submission script…

```
#!/bin/bash
#PBS -N gpuTest
#PBS -r n
#PBS -M taylor@hpc.ufl.edu
#PBS -m abe
#PBS -o gpu.log
#PBS -j oe
#PBS -l nodes=1:ppn=1:gpus=1:tesla
#PBS -l walltime=00:10:00
#PBS -q gpu
#
cd $PBS_O_WORKDIR
echo "PBS_GPUFILE…"
cat $PBS_GPUFILE
gpuNum=`cat $PBS_GPUFILE | sed -e 's/.*-gpu//g'`
echo gpuNum = $gpuNum
module load cuda
EXE=/opt/cuda-sdk/C/bin/linux/release/deviceQuery
$EXE --device=$gpuNum > deviceQuery.out 2>&1
```


UF Research Computing

- What about MPI Jobs?
  - Multiple Hosts?
  - Multiple GPUs
  - Must tell each MPI process which GPU to use
- pbsgpu-wrapper
  - Extracts information from PBS_GPUFILE
  - Sets CUDA_VISIBLE_DEVICES for each MPI process (rank)
  - Can be used with serial jobs as well

UF Research Computing

- Must Use the Assigned GPU
  - In a MPI submission script…

```
#!/bin/bash
#PBS -m abe
#PBS –M taylor@hpc.ufl.edu
#PBS -l walltime=1:00:00
#PBS -l pmem=1500mb
#PBS -N amber
#PBS -o stdout
#PBS -e stderr
#PBS -q gpu
#PBS -l nodes=2:ppn=1:gpus=1:fermi
#
module load intel openmpi cuda amber/12
export PMEMD=pmemd.cuda.MPI
cd $PBS_O_WORKDIR
echo "cat $PBS_GPUFILE"
cat $PBS_GPUFILE
export PBS_GPUS='' gpus=; while read gpu ; do gpus="$gpus,$gpu"; done < $PBS_GPUFILE; echo $gpus | sed -e 's%^,%%'`
mpiexec /usr/local/bin/pbsgpu-wrapper $PMEMD -O -i mdin -o mdout -p prmtop -c inpcrd
exit 0
```


UF Research Computing

Questions?