The OSU College of Engineering DGX System
for Advanced GPU Computing

Mike Bailey
mjb@cs.oregonstate.edu

OSU’s College of Engineering has six Nvidia DGX-2 systems

Each DGX server:
• Has 16 NVidia Tesla V100 GPUs
• Has 28TB of disk, all SSD
• Has two 24-core Intel Xeon 8168 Platinum 2.7GHz CPUs
• Has 1.5TB of DDR4-2666 System Memory
• Runs the CentOS 7 Linux operating system

Overall compute power:
• Each V100 NVidia Tesla card has 5,120 CUDA Cores and 640 Tensor Cores
• This gives each 16-V100 DGX server a total of 81,920 CUDA cores and 10,240 Tensor cores
• This gives the entire 6-DGX package a total of 491,520 CUDA Cores and 61,440 Tensor Cores
Performance Comparison with one of our other Systems

BTW, you can also use the rabbit machine:

```
ssh rabbit.engr.oregonstate.edu
```

It is a good place to write your code and get it to compile. It is **not** a good place to do the final run of your code.

---

How to SSH to the DGX Systems

```
flip3 151%  ssh  submit-c.hpc.engr.oregonstate.edu

submit-c 142%  module  load  slurm
```

Type this right away to set your path correctly.
How to Check on the DGX Systems

Check on the queues

<table>
<thead>
<tr>
<th>JOBID</th>
<th>PARTITION</th>
<th>NAME</th>
<th>USER</th>
<th>ST</th>
<th>TIME</th>
<th>NODES</th>
<th>NODELIST(REASON)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3923</td>
<td>mime4</td>
<td>c_only</td>
<td>jayasurw</td>
<td>R</td>
<td>1-10:32:19</td>
<td>1</td>
<td>compute-e-1</td>
</tr>
<tr>
<td>3963</td>
<td>mime4</td>
<td>2Dax</td>
<td>jayasurw</td>
<td>R</td>
<td>16:21:03</td>
<td>1</td>
<td>compute-e-2</td>
</tr>
<tr>
<td>3876</td>
<td>share</td>
<td>CHEC000__</td>
<td>chukwuk</td>
<td>R</td>
<td>1-23:36:45</td>
<td>1</td>
<td>compute-2-6</td>
</tr>
<tr>
<td>3971</td>
<td>mcrhp</td>
<td>tcah</td>
<td>dionnee</td>
<td>R</td>
<td>8:59:45</td>
<td>1</td>
<td>compute-h-8</td>
</tr>
<tr>
<td>3881</td>
<td>dgx2</td>
<td>bash</td>
<td>heli</td>
<td>R</td>
<td>1-22:50:44</td>
<td>1</td>
<td>compute-dgx2-1</td>
</tr>
<tr>
<td>3965</td>
<td>dgx2</td>
<td>bash</td>
<td>chenjui3</td>
<td>R</td>
<td>13:47:36</td>
<td>1</td>
<td>compute-dgx2-4</td>
</tr>
<tr>
<td>3645</td>
<td>dgx2</td>
<td>bash</td>
<td>mishrash</td>
<td>R</td>
<td>5-16:48:09</td>
<td>1</td>
<td>compute-dgx2-5</td>
</tr>
<tr>
<td>3585</td>
<td>dgx2</td>
<td>bash</td>
<td>azieren</td>
<td>R</td>
<td>6-17:34:00</td>
<td>1</td>
<td>compute-dgx2-3</td>
</tr>
<tr>
<td>3583</td>
<td>dgx2</td>
<td>bash</td>
<td>azieren</td>
<td>R</td>
<td>6-18:26:44</td>
<td>1</td>
<td>compute-dgx2-3</td>
</tr>
</tbody>
</table>

submit-c 144% sinfo

PARTITION AVAIL TIMELIMIT NODES STATE NODELIST
share* up 7-00:00:00 2 drain compute-4-[3-4]
share* up 7-00:00:00 1 mix compute-2-6
sharegpu up 7-00:00:00 1 mix compute-dgx2-1
sharegpu up 7-00:00:00 3 idle compute-dgx2-[2-3],compute-gpu
dgx2 up 7-00:00:00 1 drain compute-dgx2-2
dgx2 up 7-00:00:00 5 mix compute-dgx2-[1,3-6]
gpu up 7-00:00:00 2 mix compute-gpu[3-4]
gpu up 7-00:00:00 1 idle compute-gpu2
gpu up 7-00:00:00 1 down compute-gpu1
dgx up 7-00:00:00 3 mix compute-dgx2-[4-6]
dgx up 7-00:00:00 1 mix compute-dgx1-1
dgx up 7-00:00:00 2 idle compute-dgx2-[2-3]
class up 1-00:00 1 mix compute-dgx1-1
class up 1-00:00 2 idle compute-dgx2-[2-3]
eecs up 7-00:00:00 1 mix compute-2-6

System Information

Class partitions

Submitting a Non-batch Test-CUDA job to the DGX System

Create a bash shell file that looks like this

```
#!/bin/bash
/usr/local/apps/cuda/cuda-10.1/bin/nvcc -o montecarlo montecarlo.cu
./montecarlo
```

Note: A single dash (-) is used for a single character flag
A double dash (--) is used for a word (more than a single character) flag

These 2 lines are actual bash code

Our class account

Double dash

The bash script

submit-c 166% srun
-A cs475-575 -p classgputest --pty bash run.bash
srun: job 976138 queued and waiting for resources
srun: job 976138 has been allocated resources
Number of Trials = 2048, Blocksize = 8, MegaTrials/Second = 58.8235, Probability = 26.92%
submit-c 167%
Submitting a Batch Final-CUDA job to the DGX System

submit.bash:
```bash
#!/bin/bash
#SBATCH -J MonteCarlo
#SBATCH -A cs475-575
#SBATCH -p classgpufinal
#SBATCH --constraint=v100
#SBATCH --gres=gpu:1
#SBATCH -o montecarlo.out
#SBATCH -e montecarlo.err
#SBATCH --mail-type=BEGIN,END,FAIL
#SBATCH --mail-user=joeparallel@oregonstate.edu
/usr/local/apps/cuda/cuda-10.1/bin/nvcc -o montecarlo montecarlo.cu
./montecarlo
```

What is the Difference Between the Partitions `classgputest` and `classgpufinal`?

`classgputest` lets your program get into the system sooner, but it might be running alongside other jobs, so its performance might suffer. But, you don’t care because you are just compiling and debugging, not taking performance numbers for your report.

`classgpufinal` makes your program wait in line until it can get dedicated resources so that you get performance results that are much more representative of what the machine can do, and thus are worthy to be listed in your report.
Auto-Notifications via Email

#SBATCH --mail-user=joeparallel@oregonstate.edu

You don’t have to do this, but if you do, please be sure you get your own email address right!

Our IT people are getting real tired of fielding the bounced emails when people spell their own email address wrong.

What Showed up in my Email (which I spelled correctly)

<table>
<thead>
<tr>
<th>From</th>
<th>Subject</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slurm workload manager</td>
<td>Slurm Job_id=3980 Name=MatrixMul Ended, Run time 00:00:12, COMPLETED, ExitCode 0</td>
</tr>
<tr>
<td>Slurm workload manager</td>
<td>Slurm Job_id=3980 Name=MatrixMul Began, Queued time 00:00:01</td>
</tr>
</tbody>
</table>
Submitting a Loop

submitloop.bash:

```bash
#!/bin/bash
#SBATCH  -J MonteCarlo
#SBATCH  -A cs475-575
#SBATCH  -p classgpufinal
#SBATCH  --constraint=v100
#SBATCH  --gres=gpu:1
#SBATCH  -o montecarlo.out
#SBATCH  -e montecarlo.err
#SBATCH  --mail-type=BEGIN,END,FAIL
#SBATCH  --mail-user=joeparallel@oregonstate.edu

for t in 2048 8192 131072 2097152
do
  for b in 8 16 32 64 128 256
do
    /usr/local/apps/cuda/cuda-10.1/bin/nvcc -DNUMTRIALS=$t -DBLOCKSIZE=$b -o montecarlo montecarlo.cu
    ./montecarlo
done
done
```

These 8 lines are actual bash code

submit-c 153%
sbatch submitloop.bash
Submitted batch job 475

submit-c 154%
tail -f montecarlo.err
Displays the latest output added to montecarlo.err
Keeps doing it forever.
Control-c to get out of it.

Use slurm’s `scancel` if your Job Needs to Be Killed

submit-c 163%
sbatch submitloop.bash
Submitted batch job 476

submit-c 164%
scancel 476
Submitting an OpenCL job to the DGX System

submit.bash:

```bash
#!/bin/bash
#SBATCH  -J  PrintInfo
#SBATCH  -A  cs475-575
#SBATCH  -p  classgpufinal
#SBATCH  --constraint=v100
#SBATCH  --gres=gpu:1
#SBATCH  --output=printinfo.out
#SBATCH  --error=printinfo.err
#SBATCH  --mail-type=BEGIN,END,FAIL
#SBATCH  --mail-user=joeparallel@oregonstate.edu

g++ -o printinfo printinfo.cpp /usr/local/apps/cuda/cuda-10.1/lib64/libOpenCL.so.1.1 -lm -fopenmp
./printinfo
```

Here's what `printinfo` got on one graphics card on the DGX System

Number of Platforms = 1
Platform #0:
  Name    = 'NVIDIA CUDA'
  Vendor  = 'NVIDIA Corporation'
  Version = OpenCL 1.2 CUDA 11.2.153'
  Profile = 'FULL_PROFILE'
  Number of Devices = 1
Device #0:
  Type = 0x0004 = CL_DEVICE_TYPE_GPU
  Device Vendor ID = 0x10de (NVIDIA)
  Compute Units = 80
  Device Maximum Work Item Dimensions = 3
  Device Maximum Work Item Sizes = 1024 x 1024 x 64
  Device Maximum Work Group Size = 1024
  Device Maximum Clock Frequency = 1530 MHz

Device Extensions:
  cl_khr_global_int32_base_atomics
  cl_khr_global_int32_extended_atomics
  cl_khr_local_int32_base_atomics
  cl_khr_local_int32_extended_atomics
  cl_khr_fp64
  cl_khr_byte_addressable_store
  cl_khr_ict
  cl_khr_gl_sharing
  cl_nv_compiler_options
  cl_nv_device_attribute_query
  cl_nvPragma_unroll
  cl_nvCopy_opts
  cl_nvCreate_buffer

For comparison, rabbit's graphics card has 15 Compute Units