The OSU College of Engineering DGX System for Advanced GPU Computing

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 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 previous 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

<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_CIRC</td>
<td>chukwok</td>
<td>R</td>
<td>1-23:36:45</td>
<td>1</td>
<td>compute-2-6</td>
<td></td>
</tr>
<tr>
<td>3971</td>
<td>narhp</td>
<td>tcsh</td>
<td>dionne</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>chenju3</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>mish rash</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 143% squeue

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

submit-c 144% sinfo

Submitting a CUDA job to the DGX Systems using Slurm

Create a bash shell file that looks like this

submit.bash:

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

submit-c 143% sbatch submit.bash
Submitted batch job 474

submit-c 144% cat arraymul.err

Check the output
(I like sending my output to standard error, not standard output)
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  ArrayMul
#SBATCH  -A  cs475-575
#SBATCH  -p  class
#SBATCH  --constraint=v100
#SBATCH  --gres=gpu:1
#SBATCH  -o  arraymul.out
#SBATCH  -e  arraymul.err
#SBATCH  --mail-type=BEGIN,END,FAIL
#SBATCH  --mail-user=joeparallel@oregonstate.edu

for t in 1 2 4 8 16 32
do
  /usr/local/apps/cuda/cuda-10.1/bin/nvcc -DNUMT=$t  -o arrayMul arrayMul.cu
  ./arrayMul
done
```

These 5 lines are actual bash code

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

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

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

submit.bash:

```bash
#!/bin/bash
#SBATCH  -J  PrintInfo
#SBATCH  -A  cs475-575
#SBATCH  -p  class
#SBATCH  --constraint=v100
#SBATCH  --gres=gpu:1
#SBATCH  -o  printinfo.out
#SBATCH  -e  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

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

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_icd
- cl_khr_gl_sharing
- cl_nv_compiler_options
- cl_nv_device_attribute_query
- cl_nvPragma_unroll
- cl_nvCopy_opt
- cl_nvCreate_buffer

For reference, rabbit’s graphics card has 15 Compute Units