OSU’s College of Engineering bought 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 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 run your code.
How to SSH to the DGX Systems

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

`submit-c 142% module load slurm`

ssh over to a DGX submission machine -- `submit-a` and `submit-b` will also work

Type this right away to set your path correctly
To check on the DGX systems, you can use the `squeue` command to see the status of submitted jobs and the `sinfo` command to view the system information.

<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>2Dex</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>CH3COOH_</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>nerhp</td>
<td>tcsh</td>
<td>dionnec</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>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>

For system information, use the `sinfo` command as shown:

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

This information helps you understand the status of the queues and partitions on the DGX systems.
Submitting a CUDA job to the DGX Systems using Slurm

Create a shell file

submit.bash:

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

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

The Job Name

Your class account

This is the partition name that we use for classes

Double dash

bash code

Submit the job described in your shell file

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

Submit-c 144% cat matrixmul.err

Check the output
(I like sending my output to standard error, not standard output)
## What Showed up in my Email

<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:

#!/bin/bash
#SBATCH -J MatrixMul
#SBATCH -A cs475-575
#SBATCH -p class
#SBATCH --gres=gpu:1
#SBATCH -o matrixmul.out
#SBATCH -e matrixmul.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 matrixMul matrixMul.cu
   ./matrixMul
done

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

submit-c 154% tail -f matrixmul.err
Displays the latest output added to matrixmul.err.
Keeps doing it forever.

Control-c to get out of it.
Results for Multiplying two 1024x1024 Matrices

GigaFlops during Matrix Multiplication

GigaFlops vs NUMT

(A CUDA block was actually $NUMT \times NUMT$ threads)
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
```
submit.bash:

#!/bin/bash
#SBATCH -J MatrixMult
#SBATCH -A cs475-575
#SBATCH -p class
#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 the DGX System

Number of Platforms = 1

**Platform #0:**
- Name = 'NVIDIA CUDA'
- Vendor = 'NVIDIA Corporation'
- Version = 'OpenCL 1.2 CUDA 10.1.351'
- Profile = 'FULL_PROFILE'
- Number of Devices = 1

**Device #0:**
- Type = 0x0004 = CL_DEVICE_TYPE_GPU
- Device Vendor ID = 0x10de (NVIDIA)
- Device Maximum 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_icd
- cl_khr_gl_sharing
- cl_nv_compiler_options
- cl_nv_device_attribute_query
- cl_nv pragma unroll
- cl_nv_copy_opts
- cl_nv_create_buffer