The OSU College of Engineering DGX System for Advanced GPU Computing

Oregon State University
Mike Bailey
mjb@cs.oregonstate.edu

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License

dgx_system.pptx

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 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

submit-c 143% squeue
JOBID PARTITION     NAME    USER ST    TIME      NODES NODELIST(REASON)
3923     mime4   c_only jayasurw R 1-10:32:19      1 compute-e-1
3963     mime4     2Dex jayasurw R   16:21:03      1 compute-e-2
3876     share CH3COOH_  chukwuk R 1-23:36:45      1 compute-2-6
3971     nerhp tcsh dionnec R    8:59:45      1 compute-h-8
3881      dgx2     bash     heli R 1-22:50:44      1 compute-dgx2-1
3965      dgx2     bash  chenju3  R   13:47:36      1 compute-dgx2-4
3645      dgx2     bash mishrash R 5-16:48:09      1 compute-dgx2-5
3585      dgx2     bash  azieren R 6-17:34:00      1 compute-dgx2-3
3583      dgx2     bash  azieren R 6-18:26:44      1 compute-dgx2-3

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-dgxs-1
sharegpu up 7-00:00:00      3   idle compute-dgxs-[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]
dgx2         up 7-00:00:00      5    mix compute-gpu[3-4]
dgx2         up 7-00:00:00      1   idle compute-gpu1
dgx2         up 7-00:00:00      3    mix compute-dgx2-[4-6]
dgxs up 7-00:00:00      1    mix compute-dgxs-1
dgxs up 7-00:00:00      2   idle compute-dgxs-[2-3]
class        up    1:00:00      1    mix compute-dgxs-1
class        up    1:00:00      2   idle compute-dgxs-[2-3]
eecs up 7-00:00:00      1    mix compute-2-6

Submitting a CUDA job to the DGX Systems using Slurm

submit-c 145% sbatch submit.bash
Job has been submitted to batch queue submit-c

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

bash code
Check the output
(I like sending my output to standard error, not standard output)
Auto-Notifications via Email

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

You don’t have to do this, but if you do, please be sure you get your email address right!
The IT people are getting real tired of fielding the bounced emails when people spell their own email address wrong.

Submitting a Loop

submitloop.bash:

```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-11.1/bin/nvcc -DNUMT=$t  -o matrixMul matrixMul.cu
  ./matrixMul
done
```

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

submit-c 163% sbatch submitloop.bash Submitted batch job 475
submit-c 164% scancel 475

Submitting an OpenCL job to the DGX Systems using Slurm

submit.bash:

```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-11.1/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>Platform #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name: NVIDIA CUDA</td>
</tr>
<tr>
<td>Vendor: NVIDIA Corporation</td>
</tr>
<tr>
<td>Version: OpenCL 1.2 CUDA 11.2.153</td>
</tr>
<tr>
<td>Profile: FULL_PROFILE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Platform #0: NVIDIA CUDA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Devices = 1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Device #0: NVIDIA CUDA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type = CL_DEVICE_TYPE_GPU</td>
</tr>
<tr>
<td>Device Vendor ID = 0x10de (NVIDIA)</td>
</tr>
<tr>
<td>Device Maximum Compute Units = 80</td>
</tr>
<tr>
<td>Device Maximum Work Item Dimensions = 3</td>
</tr>
<tr>
<td>Device Maximum Work Item Sizes = 1024 x 1024 x 64</td>
</tr>
<tr>
<td>Device Maximum Work Group Size = 1024</td>
</tr>
<tr>
<td>Device Maximum Clock Frequency = 1530 MHz</td>
</tr>
</tbody>
</table>

For reference, rabbit's graphics card has 15 Compute Units.