The OSU College of Engineering DGX System for Advanced GPU Computing

Mike Bailey
mjb@cs.oregonstate.edu

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

The OSU 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 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 each16-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
Type this right away to set your path correctly

How to Check on the DGX Systems

submit-c 143%
squeue

The Job Name
Double dash
Create a shell file
Submit the job described in your shell file
This is the partition name that we use for classes
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
Submit a CUDA job to the DGX Systems using Slurm

submit-c 144% submit.bash
submit-c 144% cat matrixmul.err

The Job Name
Your class account
This is the partition name that we use for classes
Submit the job-described in your shell file
Check the output
Submit batch job 474
Create a shell file
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

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 1996; Name=MatrixMul started; Run-time 0:00:12; COMPLETE; ExitCode: 0</td>
</tr>
<tr>
<td>Slurm-workload manager</td>
<td>Slurm: job id 1996; Name=MatrixMul began; Queued time 08:55:30</td>
</tr>
</tbody>
</table>

Submitting a Loop

```
#!/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
```

Submitting batch job 475

```
sbatch submitloop.bash
```

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

```
sbatch 163% sbatch submitloop.bash
Submitted batch job 476
sbatch 164% scancel 476
```

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
  Vendor ID = 0x10de (NVIDIA)
  Maximum Compute Units = 80
  Maximum Work Item Dimensions = 3
  Maximum Work Item Sizes = 1024 x 1024 x 64
  Maximum Work Group Size = 1024
  Maximum Clock Frequency = 1530 MHz
  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

Results for Multiplying two 1024x1024 Matrices

```
import matplotlib.pyplot as plt
import numpy as np

NUMT = np.linspace(1, 32, 32)  # NUMT values
GigaFlops = np.linspace(0, 200, 200)  # GigaFlops values

for t in NUMT:
    # Run the loop for each NUMT value

plt.plot(NUMT, GigaFlops, 'o-')
plt.xlabel('NUMT')
plt.ylabel('GigaFlops during Matrix Multiplication')
plt.title('Results for Multiplying two 1024x1024 Matrices')
plt.grid(True)
plt.show()
```

Submitting an OpenCL job to the DGX Systems using Slurm

```
#submit.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
```

bash code

```
dsbatch 153% sbatch submitloop.bash
Submitted batch job 475
```

```
dsbatch 154% tail -f matrixmul.err
```

Displays the latest output added to matrixmul.err.
Keeps doing it forever.
Control-c to get out of it.