GPU

Job Submission Example

In order to submit a job to a GPU server, you need to use the `gpu` queue and specify the number of GPU cards you wish to use. The following is a job script example for running GROMACS accelerated with GPU:

```
#!/bin/bash

#$ -M netid@nd.edu   # Email address for job notification
#$ -m abe            # Send mail when job begins, ends and aborts
#$ -pe smp 1         # Specify parallel environment and legal core size
#$ -q gpu            # Run on the GPU cluster
#$ -l gpu_card=1     # Run on 1 GPU card
#$ -N job_name       # Specify job name

module load gromacs  # Required modules

export OMP_NUM_THREADS=$NSLOTS
gmx mdrun -ntomp $OMP_NUM_THREADS -nb gpu -pin on -v -s input.tpr # Run with 16 MPI tasks
and 1 GPU devices
```

Note

- If the `-pe` parallel environment is not defined in the job script, the default value is `smp 1`. Please always make sure to request enough cores for your GPU jobs.
- Please note that the runtime limit for GPU systems is 4 days.
- **Each job must have at least 1 GPU and 1 core to run.**

Installing Software on a GPU machine

In some cases, it is necessary to use a GPU server to install the software you wish to use for your GPU jobs.

Note

- Please note that the CRC does not provide any front end machines with GPUs.

For the installation an interactive session is necessary on a GPU node. The following is an example for starting an interactive session on a GPU system with 1 GPU card and 1 core:

```
qrsh -q gpu -l gpu_card=1 -pe smp 1
```

Once the connection is established, the required software may be installed.

Note

- If your research lab or faculty advisor has purchased a machine(s), there is most likely a host group you can target. For the installation you can target GPUs in a specific host group by using the `gpu@@hostgroupname` queue.
- Before installing and using a software for GPU jobs, please make sure that the software can take advantage of GPUs.

CUDA And cuDNN Modules' Avalability

Usually, for the installation the **CUDA** (Compute Unified Device Architecture) library is necessary and in some cases the **cuDNN** (CUDA for Deep Neural Network) library too. Many versions of these libraries are available on the CRC system:

```
$ module avail cuda
-------------------------------------
/afs/crc.nd.edu/x86_64_linux/Modules/modules/development_tools_and_libraries ------------
------------------------
cuda/10.0  cuda/10.2  cuda/11.0  cuda/11.2  cuda/11.6  cuda/11.8  cuda/12.1
$ module avail cudnn
-------------------------------------
/afs/crc.nd.edu/x86_64_linux/Modules/modules/development_tools_and_libraries ------------
------------------------
cudnn/7.4  cudnn/8.0.4  cudnn/8.9.3  cudnn/v7.0
```

Note

- If you wish to use a CUDA/cuDNN version which is not installed on the CRC system, you may install other versions with Conda.

Available Hardware For General Access

You can find a list of the CRC owned GPU systems on Available Hardware.

Note

- These machines have typically 32 cores and 4 GPUs per node.
- Each GPU has an ID within the machine, this ID can be 0, 1, 2 or 3.

Resource and Job Monitoring

You can monitor the status and availability of GPU resources with the `free_gpus.sh` script.

```
free_gpus.sh @crc_gpu        # For general access
free_gpus.sh @crc_a10        # You can target host groups
```

The Xymon monitoring system can be used to analyze the behavior of processes on a given GPU machine.

You can check your job's GPU usage. In order to do that, knowing the GPU ID is necessary. You can check the ID with the following command:

```
qstat -j jobID
```

You can find it under the `resource map`. Here is an example, the GPU ID is the number in brackets:

```
resource map          1:    gpu_card=qa-a10-023.crc.nd.edu=(1)
```

Other Resources For GPU Jobs

If you wish to run large number of GPU jobs, you may want to consider submitting your jobs via Condor. You can find detailed documentation and examples on HTCondor.

If you wish to use GPUs for Machine Learning, you may want to consider using CAML ND. You can find detailed documentation and examples on CAML.