Submitting CUDA or OpenACC Jobs

GPUs available

Crane has four types of GPUs available in the **gpu** partition. The type of GPU is configured as a SLURM feature, so you can specify a type of GPU in your job resource requirements if necessary.

<table>
<thead>
<tr>
<th>Description</th>
<th>SLURM feature</th>
<th>Available Hardware</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tesla M2070, non-IB</td>
<td>gpu_m2070</td>
<td>3 nodes - 2 GPUs per node</td>
</tr>
<tr>
<td>Tesla K20, with IB</td>
<td>gpu_k20</td>
<td>3 nodes - 3 GPUs per node</td>
</tr>
<tr>
<td>Tesla K40, with IB</td>
<td>gpu_k40</td>
<td>5 nodes - 4 K40M GPUs per node</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 node - 2 K40C GPUs per node</td>
</tr>
<tr>
<td>Tesla P100, with OPA</td>
<td>gpu_p100</td>
<td>2 nodes - 2 GPUs per node</td>
</tr>
</tbody>
</table>

To run your job on the next available GPU regardless of type, add the following options to your `srun` or `sbatch` command:

```
--partition=gpu --gres=gpu
```

To run on a specific type of GPU, you can constrain your job to require a feature. To run on K40 GPUs for example:

```
--partition=gpu --gres=gpu --constraint=gpu_k40
```

You may request multiple GPUs by changing the `--gres` value to `--gres=gpu:2`. Note that this value is **per node**. For example, `--nodes=2 --gres=gpu:2` will request 2 nodes with 2 GPUs each, for a total of 4 GPUs.

Compiling

Compilation of CUDA or OpenACC jobs must be performed on the GPU nodes. Therefore, you must run an **interactive job** to compile. An example command to compile in the **gpu** partition could be:

```
$ srun --partition=gpu --gres=gpu --mem-per-cpu=1024 --ntasks-per-node=6 --nodes=1 --pty $SHELL
```

The above command will start a shell on a GPU node with 6 cores and 6GB of ram in order to compile a GPU job. The above command could also be useful if you want to run a test GPU job interactively.

Submitting Jobs

CUDA and OpenACC submissions require running on GPU nodes.
OpenACC submissions require loading the PGI compiler (which is currently required to compile as well).

```
openacc.submit

#!/bin/sh
#SBATCH --time=03:15:00
#SBATCH --mem-per-cpu=1024
#SBATCH --job-name=cuda-acc
#SBATCH --partition=gpu
#SBATCH --gres=gpu
#SBATCH --error=/work/[groupname]/[username]/job.%J.err
#SBATCH --output=/work/[groupname]/[username]/job.%J.out

module load cuda/6.5 compiler/pgi/16
./acc-app.exe
```