GPU Nodes

Puma

Puma has a different arrangement for GPU nodes than Ocelote and ElGato. Whereas the older clusters have one GPU per node, Puma has four. This has a financial advantage for providing GPU's with lower overall cost, and a technical advantage of allowing jobs that can use multiple GPU's to run faster than spanning multiple nodes. This capability comes from using a newer operating system.
Each node has four Nvidia V100S model GPUs. They are provisioned with 32GB memory compared to 16GB on the P100's.
Running jobs. SLURM information.

	V100 PCIe	V100 SXM2	V100S PCIe
GPU Architecture	NVIDIA Volta		
NVIDIA Tensor Cores	640		
NVIDIA CUDA® Cores	5,120		
Double-Precision Performance	7 TFLOPS	7.8 TFLOPS	8.2 TFLOPS
Single-Precision Performance	14 TFLOPS	15.7 TFLOPS	16.4 TFLOPS
Tensor Performance	112 TFLOPS	125 TFLOPS	130 TFLOPS
GPU Memory	32 GB /16 GB HBM2		32 GB HBM2
Memory Bandwidth	900 GB/sec		1134 GB/sec
ECC	Yes		

Ocelote

Ocelote has 45 compute nodes with Nvidia P100 GPUs. These are available to researchers on campus. The limitation is a maximum of 10 concurrent jobs.
Ocelote also has one node with a V100. Since there is only one, you can feel free to use it for testing and comparisons to the P100. But production work should be run on the P100's.
Ocelote has one node with two P100's for testing jobs that use two GPU's. This one should be used to compare with running a job on two nodes. Use *np100s=2*.
Ocelote is built on CentOS 6 which is the foundation for all compilers, libraries, and applications. Some workloads need CentOS 7 for its newer kernel. A number of GPU nodes have been reprovisioned with a CentOS 7 image. PBS information

Ocelote is scheduled to be rebuilt in mid-2021 when its 5 year support concludes. It is likely to resemble Puma with Slurm and Centos7

PERFORMANCE SPECIFICATION FOR NVIDIA TESLA P100 ACCELERATORS

	P100 for PCIe-Based Servers
Double-Precision Performance	4.7 TeraFLOPS
Single-Precision Performance	9.3 TeraFLOPS
Half-Precision Performance	18.7 TeraFLOPS
NVIDIA NVLink™ Interconnect Bandwidth	-
PCIe x16 Interconnect Bandwidth	32 GB/s
CoWoS HBM2 Stacked Memory Capacity	16 GB or 12 GB
CoWoS HBM2 Stacked Memory Bandwidth	732 GB/s or 549 GB/s
Enhanced Programmability with Page Migration Engine	✓
ECC Protection for Reliability	✓

Cuda Modules

The latest Cuda module available on Puma is 11.0 and is the only version until newer ones come along.
The Cuda driver version on each V100S can be queried with the *nvidia-smi* command.
module load cuda will load the latest version of cuda.
module list will show what you have loaded.
module load cuda-sdk will load the Software Developers Kit with compilers, libraries and the profiler.
module load cuda-dnn will load the Nvidia Deep Learning Network library. It is separate due to licensing needs.

The latest Cuda module available on Ocelote is 10.1 Prior versions are still available. Newer versions are not possible until Ocelote is rebuilt.
The Cuda driver version on each P100 can be queried with the *nvidia-smi* command.
module load cuda will load cuda10.1/toolkit/10.1.168
You may also need *module load cuda101/neuralnet* in case you need the cuDNN libraries for Tensorflow
module load tensorrt will provide support for tensorflow, particularly the library called libnvinfer.so.6

Nvidia Nsight Compute (the interactive kernel profiler) is not available on Ocelote's P100 GPUs at all. In response to a security alert (CVE-2018-6260) this capability is only available with root authority which users do not have. So it is not available on Puma either.

OpenACC

The OpenACC API is a collection of compiler directives and runtime routines that allow you to specify loops and regions of code in standard C, C++, and Fortran that you can offload from a host CPU to the GPU.

We provide two methods of support for OpenACC

1. We support OpenACC in the PGI Compiler. The PGI implementation of OpenACC is considered the best implementation.
   "module load pgi" on Ocelote. If you are on a GPU node from an interactive session you can run "pgaccelinfo" to test functionality. Remember that the login nodes do not have GPUs installed.
   A useful getting started guide written by Nvidia is at:
   https://www.pgroup.com/doc/openacc17_gs.pdf
2. We support OpenACC in the GCC Compiler 6.1 which is automatically loaded as a module when you log into Ocelote. Verify with "module list".
   The GCC 6 release includes a much improved implementation of the OpenACC 2.0a specification.

A useful quick reference guide can be found at:
https://gcc.gnu.org/wiki/OpenACC#Quick_Reference_Guide

About two times a year we host the Xsede Workshop on Programming GPU's with OpenACC. These courses provide an overview of how to accelerate your code without a lot of programming knowledge. Watch for announcements to the HPC-Info list.
https://www.psc.edu/xsede-hpc-series-all-workshops

Nvidia has available free online OpenACC courses:
https://developer.nvidia.com/openacc/overview
https://developer.nvidia.com/openacc-courses

Applications

Many applications have been optimized to run faster on GPU's. These include:

- NAMD - installed as a module; *module load namd_cuda*
- VASP - A restricted license version is installed on Puma/Ocelote; only available to the licensed users
- GROMACS - Installed as a module on Puma/Ocelote; *module load gromacs*
- LAMMPS - Installed as a module on Puma/Ocelote; *module load lammps*
- ABAQUS - Installed as a module on Puma/Ocelote; *module load abaqus*
- GAUSSIAN - Installed as a module on Puma/Ocelote; *module load gaussian/g16.* See these notes
- MATLAB - Review the GPU Coder at their web site
- ANSYS Fluent
- RELION - available as a Singularity container or as a module.
- ML and DL frameworks - See the next section below

NVIDIA GPU Cloud Container Registry

We support the use of HPC and ML/DL containers available on NVIDIA GPU Cloud (NGC). Many of the popular HPC applications including NAMD, LAMMPS and GROMACS containers are optimized for performance and available to run in Singularity on Ocelote or Puma. On Ocelote these containers must be run on nodes with CentOS 7 installed.

The containers and respective README files can be found at /unsupported/singularity/nvidia/os7 on Ocelote.
The containers and respective README files can be found at /contrib/singularity/nvidia on Puma.

The other option for accessing ML / DL frameworks is through Python 3.6 on Ocelote and Python 3.8 on Puma. This is covered in the next section.

The Nvidia images have been modified to include bindings for your /extra and /groups directories if you want to run your jobs from those directories.

These containers have not been updated in a while for two reasons; first that the newer versions would only run on nodes built with CentOS 7, and second that most of them are available in Python 3.6. See elsewhere on this page for the usage of CentOS 7 nodes.

nvidia-caffe.18.09-py2.simg	Caffe is a deep learning framework made with expression, speed, and modularity in mind. It was originally developed by the Berkeley Vision and Learning Center (BVLC)
nvidia-pytorch.18.09-py3.simg	PyTorch is a Python package that provides two high-level features: • Tensor computation (like numpy) with strong GPU acceleration • Deep Neural Networks built on a tape-based autograd system
nvidia-mxnet.18.09.simg	MXNet is a deep learning framework designed for both efficiency and flexibility. It allows you to mix the flavors of symbolic programming and imperative programming to maximize efficiency and productivity.
nvidia-tensorflow.18.09-py3.simg	TensorFlow is an open source software library for numerical computation using data flow graphs. TensorFlow was originally developed by researchers and engineers working on the Google Brain team within Google's Machine Intelligence research organization for the purposes of conducting machine learning and deep neural networks research.
nvidia-theano.18.08.simg	Theano is a Python library that allows you to define, optimize, and evaluate mathematical expressions involving multi-dimensional arrays efficiently.

Each is provided in a Singularity container.

The file name has a tag at the end that represents when it was made, so 18.01 is January 2018

Pulling Nvidia ML / DL Images on Ocelote

It is possible for you to create your own Singularity containers on Ocelote pulling down the images created by Nvidia. The general rule that you cannot create your own containers because that would require root authority still applies. Root authority is not required if you follow this procedure.

Follow this procedure.

Python ML / DL including Nvidia RAPIDS

Nvidia Rapids is only available on Ocelote currently.
Tensorflow and tensorboard are available on both Puma (python/3.8) and Ocelote (ocelote/3.6)
The minimum version of Python that is supported is 3.6 so you will module load python/3.6 on Ocelote, for all these functions. This will get you to installed packages including:

Framework	Details
numba	RAPIDS: numba is for Cuda programming
cuml	RAPIDS: Cuda Machine Learning has many ML algorithms like K-means, PCA and SVM
cudf	RAPIDS: Cuda Dataframes supports loading and manipulating datasets
tensorflow	TensorFlow is an open source software library for numerical computation using data flow graphs.
torch	PyTorch supports tensor computation and deep neural networks
caffe2	A deep learning framework
tensorrt	Inference server for deep learning
tensorboard	Visualization tool for machine learning

You must use a node with CentOS 7 as mentioned above, and you may need to load modules beyond python/3.6, such as tensorrt, cuda101/neuralnet

PBS Usage including CentOS 7 (Ocelote)

The GPU nodes have more memory than the other Ocelote nodes so the *select* statements reflect 8GB per core by 28 cores equals 224GB. The following examples use the CentOS 7 nodes with the attribute *os7=True*

Either: copy the file you wish to use to your directory. Your home path as well as /extra and /xdisk have been bound to the image, so those are your choices.

Or: run the singularity file from where it is. Since you cannot modify it you will not interfere with anyone else.

For interactive use, start an interactive job on a GPU node modifying this command:

```
$ qsub -I -N jobname -W group_list=GROUP-NAME -q windfall -l select=1:ncpus=28:mem=224gb:np100s=1:os7=True -
l walltime=1:0:0
```

You must change the group_list and you should change the other attributes as desired.

On the compute node assigned to you, as an example you can run:

```
$ module load singularity
$ singularity exec --nv nvidia-tensorflow.18.01-py3.simg python tensorflow_example.py
```

You need to include the --nv and note it has two dashes. This will bind the Cuda libraries.
The example file is included in this directory. "tensorflow_example.py"

For batch use, you will include these three lines in your submission script

```
#PBS -l select=1:ncpus=28:mem=224gb:np100s=1:os7=True
module load singularity
singularity exec --nv nvidia-tensorflow.18.01-py3.simg python tensorflow_example.py
```

You will want exclusive access to the node so there is not contention for the GPU. That is obtained by asking for all 28 cores as shown above

Note: the option **np100s=0** is not valid. Just leave out that part of the statement if you are not using GPU's

The test V100 node will have **nv100s=1** instead of **np100s=1.**
#QSUB -l select=1:ncpus=28:mem=224gb:np100s=1:os7=True

There are more detailed examples here

Singularity

For more information on Singularity, see their web site at:

http://singularity.lbl.gov/user-guide

There are tutorials for Singularity on HPC here

Training

We host workshops from the Pittsburgh Supercomputer Center which is a NSF funded location. We are working with Nvidia to offer a workshop in the April 2018 timeframe.

Watch for announcements from the hpc-info list.

Nvidia periodically runs training sessions like these ones:
Accelerate Your Code with OpenACC