December User Conference Call
Analysis Analysis and Visualization Tools at OLCF

Benjamin Hernandez
Computer Scientist
Analytics and AI Methods at Scale Group
OLCF

Dec 8th, 2021
Contents

- General overview of analysis infrastructure at OLCF

- Visualization Tools
  - Remote Desktops
    - TurboVNC and NiceDCV

- Scientific Visualization Tools Overview
  - Paraview
  - Visit
Tools Overview

- docs.olcf.ornl.gov
Tools Overview. Python at OLCF

Python on OLCF Systems

Overview

In high-performance computing, Python is heavily used to analyze scientific data on the system. Some users require specific versions of Python or niche scientific packages to analyze their data, which may further depend on numerous other Python packages. Because of all the dependencies that some Python packages require, and all the types of data that exist, it can be quite troublesome to get different Python installations to "play nicely" with each other, especially on an HPC system where the system environment is complicated. Conda, a package and virtual environment manager from the Anaconda distribution, helps alleviate these issues.

Conda allows users to easily install different versions of binary software packages and any required libraries appropriate for their computing platform. The versatility of conda allows a user to essentially build their own isolated Python environment, without having to worry about clashing dependencies and other system installations of Python. Conda is available on OLCF systems, and loading the default Python module loads an Anaconda Python distribution. Loading this distribution automatically puts you in a "base" conda environment, which already includes packages that one can use for simulation, analysis, and machine learning.

For users interested in using Python with Jupyter, see Jupyter at OLCF instead.

OLCF Python Guides

Below is a list of guides created for using Python on OLCF systems.

- Conda Basics Guide: Goes over the basic workflow and commands of Conda
- Installing Parallel h5py Guide: Teaches you how to install parallel-enabled h5py and mpi4py
- Installing CuPy Guide: Teaches you how to install CuPy
ML/DL & Data Analytics

There are several options for various kinds of machine learning, deep learning, and data analytics tasks on OLCF systems.

- IBM Watson Machine Learning CE -> Open CE
  - Getting Started
  - Running Distributed Deep Learning Jobs
  - Setting up Custom Environments
  - Best Distributed Deep Learning Performance
  - Example

- R and pbDr on Summit
  - Loading R
  - How to Run an R Script
  - R Hello World Example
  - pbDr Hello World Example
  - Common R Packages for Parallelism
  - GPU Computing with R
  - More Information

- NVIDIA RAPIDS
  - Overview
  - Getting Started
  - RAPIDS on Jupyter
  - RAPIDS on Summit
  - Setting up Custom Environments
  - BlazingSQL Distributed Execution
Jupyter at OLCF
Visualization tools

Remote Visualization using Nice DCV (GPU nodes only)

Step 1 (terminal 1)
Launch an interactive job:

```bash
local:ssh username@andes.olcf.ornl.gov
andes: salloc -A PROJECT_ID -p gpu -N 1 -t 60:00 -M andes --constraint=DCV
```
Run the following commands:

```bash
$ xinit &
$ export DISPLAY=:0
$ dcv create-session --gl-display :0 mySessionName
$ hostname // will be used to open a tunneling connection with this node
$ andes-gpuN
```

Step 2 (terminal 2)
Open a tunneling connection with gpu node N, given by hostname:

```bash
local:ssh username@andes.olcf.ornl.gov -L 8443:andes-gpuN:8443
```
Open your web browser using the following link and use your credentials to access OLCF systems:

https://localhost:8443 When finished, kill the dcv session in first terminal:

```bash
$ dcv close-session mySessionName
$ kill %
```

Previous
Remote Desktops. TurboVNC and NiceDCV

- Users can access and control a remote desktop (xfce) running on Andes.
  - Useful when a viz. tool does not have a client/server architecture as in Paraview or Visit

- TurboVNC and NiceDCV as a low-latency alternative to X-forwarding
  - RFB (remote frame buffer protocol) for optimal keyboard and mouse event and frame buffer delivery.
  - Low latency encoding
    - TurboVNC - JPEG encoding on CPU's SIMD instruction set
    - NiceDCV - H264 encoding using NVENC on GPU
Remote Desktops.

<table>
<thead>
<tr>
<th></th>
<th>TurboVNC</th>
<th>TurboVNC / VirtualGL</th>
<th>NiceDCV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open source</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Availability</td>
<td>Regular nodes</td>
<td>GPU nodes</td>
<td>GPU Nodes (5 seats)</td>
</tr>
<tr>
<td>Use case</td>
<td>Non graphics intensive apps, e.g. 2D graphics, any user interface (matlab, performance tools, editors, etc.)</td>
<td>GPU accelerated 3D graphics, apps with no client/ server architecture e.g. VMD, yt’s 3D visualization, USC Chimera, custom viz. tools CUDA+OpenGL</td>
<td>GPU accelerated 3D graphics, apps with no client/ server architecture e.g. VMD, yt’s 3D visualization, USC Chimera, custom viz. tools CUDA+OpenGL</td>
</tr>
<tr>
<td>Compression</td>
<td>TurboJPEG (SIMD) • Configurable Quality / Compression settings</td>
<td>TurboJPEG (SIMD) • Configurable Quality / Compression settings</td>
<td>H264 (NVENC) • Configurable Quality / Compression settings</td>
</tr>
<tr>
<td>Client</td>
<td>vncviewer</td>
<td>vncviewer</td>
<td>Web browser</td>
</tr>
</tbody>
</table>
Visualization Tools

- **Visit**
  - Overview
  - Installing and Setting Up Visit
  - Remote GUI Usage
  - Command Line Example
  - Troubleshooting
  - Additional Resources

- **ParaView**
  - Overview
  - Installing and Setting Up ParaView
  - Remote GUI Usage
  - Command Line Example
  - Troubleshooting
  - Additional Resources
Thanks!

Issues and feedback

help@olcf.ornl.gov