Argonne Training Program on Extreme-Scale Computing (ATPESC)

Data Analysis and Visualization
## Visualization & Data Analysis

<table>
<thead>
<tr>
<th>Time</th>
<th>Title of presentation</th>
<th>Lecturer</th>
</tr>
</thead>
<tbody>
<tr>
<td>9:30 am</td>
<td>Data Analysis and Visualization Introduction</td>
<td>Mike Papka ANL/NIU, Joe Insley ANL/NIU, Silvio Rizzi, ANL</td>
</tr>
<tr>
<td>10:15 am</td>
<td>Scalable Molecular Visualization and Analysis Tools in VMD</td>
<td>John Stone UIUC</td>
</tr>
<tr>
<td>11:00 am</td>
<td>Break</td>
<td></td>
</tr>
<tr>
<td>11:15 am</td>
<td>Large Scale Visualization with ParaView</td>
<td>Dan Lipsa Kitware</td>
</tr>
<tr>
<td>12:30 pm</td>
<td>Lunch</td>
<td></td>
</tr>
<tr>
<td>1:30 pm</td>
<td>Visualization and Analysis of HPC Simulation Data with VisIt</td>
<td>Cyrus Harrison LLNL</td>
</tr>
<tr>
<td>2:45 pm</td>
<td>Vapor</td>
<td>Scott Pearse UCAR</td>
</tr>
<tr>
<td>3:00 pm</td>
<td>Break</td>
<td></td>
</tr>
<tr>
<td>3:45 pm</td>
<td>Exploring Visualization with Jupyter Notebooks</td>
<td>Tommy Marrinan St. Thomas / ANL, David Koop NIU, Cyrus Harrison LLNL, Matt Larsen LLNL</td>
</tr>
<tr>
<td>5:00 pm</td>
<td>Adjourn</td>
<td></td>
</tr>
</tbody>
</table>
Here’s the plan…

— Examples of visualizations
— Visualization resources
— Visualization tools and formats
— Data representations
— Visualization for debugging
— In Situ Visualization and Analysis
Multi-Scale Simulation / Visualization
Arterial Blood Flow

Data courtesy of: George Karniadakis and Leopold Grinberg, Brown University
Physics: Stellar Radiation

Data courtesy of: Lars Bildsten and Yan-Fei Jiang, University of California at Santa Barbara
Physics: Magnetic Confinement Fusion

Data courtesy of Sean Dettrick, TAE Technologies, Inc.
Materials Science / Molecular

Data courtesy of: Jeff Greeley, Nichols Romero, Argonne National Laboratory

Data courtesy of: Paul Kent, Oak Ridge National Laboratory, Anouar Benali, Argonne National Laboratory

Data courtesy of: Subramanian Sankaranarayanan, Argonne National Laboratory
Cosmology

Data courtesy of: Salman Habib, Katrin Heitmann, and the HACC team, Argonne National Laboratory
Cooley: Analytics/Visualization cluster

Peak 223 TF
126 nodes; each node has
- Two Intel Xeon E5-2620 Haswell 2.4 GHz 6-core processors
- NVIDIA Tesla K80 graphics processing unit (24GB)
- 384 GB of RAM
Aggregate RAM of 47 TB
Aggregate GPU memory of ~3TB
Cray CS System
216 port FDR IB switch with uplinks to our QDR infrastructure
Mounts the Theta, Eagle, and Grand file systems
Visualization Tools and Data Formats
All Sorts of Tools

Visualization Applications
- VisIt ∗
- ParaView ∗
- EnSight

Domain Specific
- VMD, PyMol, Ovito, Vapor

APIs
- VTK ∗: visualization
- ITK: segmentation & registration

GPU performance
- vl3: shader-based volume and particle rendering

Analysis Environments
- Matlab
- Parallel R

Utilities
- GnuPlot
- ImageMagick ∗

Available on Cooley
≠ Available on Theta
ParaView & VisIt vs. vtk

ParaView & VisIt
- General purpose visualization applications
- GUI-based
- Client / Server model to support remote visualization
- Scriptable / Extendable
- Built on top of vtk (largely)
- In situ capabilities

vtk
- Programming environment / API
- Additional capabilities, finer control
- Smaller memory footprint
- Requires more expertise (build custom applications)
## Data File Formats (ParaView & VisIt)

<table>
<thead>
<tr>
<th>Format Type</th>
<th>Data Description</th>
<th>Subformat</th>
<th>Format Type</th>
<th>Subformat</th>
</tr>
</thead>
<tbody>
<tr>
<td>VTK</td>
<td>SpyPlot CTH</td>
<td>PNG</td>
<td>VASP</td>
<td>VASP</td>
</tr>
<tr>
<td>Parallel (partitioned) VTK</td>
<td>HDF5 raw image data</td>
<td>SAF</td>
<td>ZeusMP</td>
<td>ZeusMP</td>
</tr>
<tr>
<td>VTK MultiBlock</td>
<td>DEM</td>
<td>LS-Dyna</td>
<td>ANALYZE</td>
<td>ANALYZE</td>
</tr>
<tr>
<td>(MultiGroup, Hierarchical,</td>
<td>VRML</td>
<td>Nek5000</td>
<td>BOV</td>
<td>BOV</td>
</tr>
<tr>
<td>Hierarchical Box)</td>
<td>PLY</td>
<td>OVERFLOW</td>
<td>GMV</td>
<td>GMV</td>
</tr>
<tr>
<td>Legacy VTK</td>
<td>Polygonal Protein Data Bank</td>
<td>paraDIS</td>
<td>Tecplot</td>
<td>Tecplot</td>
</tr>
<tr>
<td>Parallel (partitioned) legacy VTK</td>
<td>XMed Molecule</td>
<td>PATRAN</td>
<td>Vis5D</td>
<td>Vis5D</td>
</tr>
<tr>
<td>EnSight files</td>
<td>Gaussian Cube</td>
<td>PFLOTRAN</td>
<td>Xmdv</td>
<td>Xmdv</td>
</tr>
<tr>
<td>EnSight Master Server</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Exodus</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BYU</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XDMF</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLOT2D</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLOT3D</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Data Representations
Data Representations: Volume Rendering
Data Representations: Glyphs

2D or 3D geometric object to represent point data
Location dictated by coordinate
– 3D location on mesh
– 2D position in table/graph
Attributes of graphical entity dictated by attributes of data
– color, size, orientation
Data Representations: Contours (Isosurfaces)

A Line (2D) or Surface (3D), representing a constant value

VisIt & ParaView:
– good at this

vtk:
– same, but again requires more effort
Data Representations: Cutting Planes

Slice a plane through the data
- Can apply additional visualization methods to resulting plane
VisIt & ParaView & vtk good at this
VMD has similar capabilities for some data formats
Data Representations: Streamlines

From vector field on a mesh (needs connectivity)
– Show the direction an element will travel in at any point in time.
VisIt & ParaView & vtk good at this
Data Representations: Pathlines

From vector field on a mesh (needs connectivity)
– Trace the path an element will travel over time.
VisIt & ParaView & vtk good at this
Molecular Dynamics Visualization

VMD:
- Lots of domain-specific representations
- Many different file formats
- Animation
- Scriptable

VisIt & ParaView:
- Limited support for these types of representations, but improving

VTK:
- Anything’s possible if you try hard enough
Visualization for Debugging
Visualization for Debugging
Visualization for Debugging
Visualization for Debugging
Visualization for Debugging
Visualization for Debugging
Visualization for Debugging
Visualization for Debugging
Visualization for Debugging
Visualization for Debugging
Visualization for Debugging
Visualization as Diagnostics: Color by Thread ID
Visualization as Diagnostics: Color by Thread ID
In Situ Visualization and Analysis
Five orders of magnitude between compute and I/O capacity on Titan Cray system at ORNL.
Five orders of magnitude between compute and I/O capacity on Titan Cray system at ORNL

Computation: 125 PB/s
Node memory: 4.5 PB/s

Image courtesy Ken Moreland
Five orders of magnitude between compute and I/O capacity on Titan Cray system at ORNL

- Computation: 125 PB/s
- Node memory: 4.5 PB/s

Image courtesy Ken Moreland
Five orders of magnitude between compute and I/O capacity on Titan Cray system at ORNL

Computation: 125 PB/s
Node memory: 4.5 PB/s
Interconnect: 24 TB/s

Image courtesy Ken Moreland
Five orders of magnitude between compute and I/O capacity on Titan Cray system at ORNL

- Computation: 125 PB/s
- Node memory: 4.5 PB/s
- Interconnect: 24 TB/s

Image courtesy Ken Moreland
Five orders of magnitude between compute and I/O capacity on Titan Cray system at ORNL.

- **Computation**: 125 PB/s
- **Node memory**: 4.5 PB/s
- **Interconnect**: 24 TB/s
- **Storage**: 1.4 TB/s

Image courtesy Ken Moreland
What are the problems?

- Not enough I/O capacity on current HPC systems, and the trend is getting worse.
- If there’s not enough I/O, you can’t write data to storage, so you can’t analyze it: lost science.
- Energy consumption: it costs a lot of power to write data to disk.
- Opportunity for doing better science (analysis) when have access to full spatiotemporal resolution data.

Slide courtesy the SENSEI team [www.sensei-insitu.org](http://www.sensei-insitu.org)
Two Frameworks for In Situ Vis and Analysis at ALCF

- “Write once, run everywhere” design
- Data model based on VTK from Kitware
- Supports a variety of backends, including ParaView/Catalyst, VisIt/LibSim, ADIOS, Python

- Flyweight design, minimizes dependencies
- Data model based on Conduit from LLNL
- Vis and analysis algorithms implemented in VTK-m
Instrumenting Simulation Codes

1. initialize sim
2. if do_insitu bridge::initialize
3. do
4.   compute new state
5.   if do_io write plot file
6. if do_insitu bridge::execute
7. while !done
8. if do_insitu bridge::finalize
9. finalize sim

```cpp
// Run Ascent
Ascent ascent;
ascent.open();
ascent.publish(data);
ascent.execute(actions);
ascent.close();
```
SENSEI + ASCENT tutorial at SC19 and SC20

Slides and Virtual Machine available here:
https://sensei-insitu.org/tutorials/sc19.html
https://ix.cs.uoregon.edu/~hank/sc20/
SENSEI + ASCENT tutorial at SC19 and SC20

Slides and Virtual Machine available here:
https://sensei-insitu.org/tutorials/sc19.html
https://ix.cs.uoregon.edu/~hank/sc20/

SENSEI + ASCENT tutorial accepted at SC21

Date and time TBD
Exascale Computing Project
Software Technology Data and Visualization

Apps

In situ Infrastructure

ALPINE

ASCENT

VisIt Libsim

ParaView Catalyst

In situ Algorithms

New Algorithms

Traditional Algorithms

Compression

ZFP

Output/Artifacts

Cinema

Traditional Output

ParaView

VisIt

Post Processing

Slide courtesy of the ECP ALPINE project
VTK-m’s main thrust: a write-once-run-everywhere framework

Contour → Streams → Clip → Render

Surface → Normals → Ghost Cells → Warp → ...

x86 → CUDA → Xeon Phi → Radeon → Xe

Demonstrated → Upcoming

Slide courtesy of the ECP VTK-m project
What is Cinema?

- **Cinema** is part of an integrated workflow, providing a method of extracting, saving, analyzing or modifying and viewing complex data artifacts from large scale simulations.
  - If you’re having difficulty exploring the complex results from your simulation, Cinema can help.

- **The Cinema ‘Ecosystem’** is an integrated set of writers, viewers, and algorithms that allow scientists to export, analyze/modify and view Cinema databases.
  - This ecosystem is embodied in widely used tools (ParaView, VisIt, Ascent) and the database specification.
Development challenges:
• Requires good understanding of VTK data model and APIs

Build/development challenges:
• Requires a CMake-based build system
• Requires ParaView SDK (cannot use distributed ParaView binaries)
• Simulation build tightly coupled with ParaView version used

Maintenance challenges:
• Changing APIs and data model
• Changing build system
Catalyst Revised: the design

Simplifying the adaptor
----> switch to Conduit
  • Avoid need to understand VTK data model
  • Provide mechanism to provide data with zero-copy & meta-data to interpret it

Simplifying build and deployment
  • Inspired by MPICH ABI compatibility initiative
  • Simulations to link against a tiny stub and allow switching of implementation at runtime

Utkarsh Ayachit, Andrew Bauer, Ben Boeckel, Berk Geveci, Ken Moreland, Patrick O’Leary, and Tom Osika: *Catalyst Revised: Rethinking the ParaView In Situ Analysis and Visualization API*, WOIV 2021
QUESTIONS?

Joe Insley
insley@anl.gov

Silvio Rizzi
srizzi@anl.gov

Janet Knowles
jknowles@anl.gov

Victor Mateevitsi
vmateevitsi@anl.gov

www.anl.gov