VTK-m

As exascale simulations generate data, scientists need to extract information and understand their results. One of the primary mechanisms for understanding these results is to produce visualizations that can be viewed and manipulated. The VTK-m project is developing and deploying scientific visualization software capable of efficiently using exascale architectural features, such as the shared-memory parallelism available on many-core CPUs and GPUs, by redeveloping, implementing, and supporting necessary visualization algorithms.

One of the biggest recent changes in high-performance computing is the increasing use of accelerators. Accelerators contain processing cores that independently are inferior to a core in a typical CPU, but these cores are replicated and grouped such that their aggregate execution provides a very high computation rate at a much lower power. Current and future CPU processors also require much more explicit parallelism. Each successive version of the hardware packs more cores into each processor, and technologies like hyperthreading and vector operations require even more parallel processing to leverage each core's full potential.

The scientific visualization community has been building scalable tools for over 15 years that enable scientists to visualize the results of their simulations. Current tools, however, are based on a message-passing programming model and rely on a coarse decomposition that is known to break down as the level of concurrency increases. The VTK-m project is providing the core capabilities to perform scientific visualization on exascale architectures, thus filling the critical feature gap of performing efficient visualization and analysis on many-core CPU and GPU architectures.

The VTK-m team is providing general-purpose scientific visualization software for exascale architectures that supports shared memory parallelism and fine-grained concurrency. The team is focused on providing abstract models for data and execution that can be applied to a variety of algorithms across many different processor architectures, along with necessary visualization algorithm implementations. The results of this project will be delivered in tools currently used around the world today, like ParaView and VisIt, as well as in a stand-alone form.

PI: Ken Moreland, Sandia National Laboratories

Collaborators: Sandia National Laboratories, Oak Ridge National Laboratory, Los Alamos National Laboratory, University of Oregon, Kitware, Inc.

Progress to date

- The ECP project has been diligently building the features of the VTK-m visualization library to include numerous visualization features including surfacing algorithms like external faces, normal generation, and contouring, multiblock and ghost cell management, geometry transformations, compression, particle advection, and a self-contained rendering library.
- The team added support to VTK-m for multiple threading libraries, including OpenMP, to better match the exascale application codes with which it integrates. The team has also tuned the performance of VTK-m on the Summit supercomputer by introducing custom kernel scheduling parameters for the hardware on that machine, which doubled (or more) the performance for many important algorithms.
- A new functionality for identifying connected components in image and mesh data was recently added. This feature has a wide application area, including image processing, computer vision, and machine learning.
- The Contour filter in VTK-m was extended to handle all 3D cell types. This enables VTK-m to create isosurface contours for unstructured grids of zoo cells in addition to meshes uniformly made of hexahedra (such as structured grids).
- In collaboration with the ECP/ALPINE and the ASC/ATDM teams, pattern recognition for image data based on Moments was added. Based on a serial algorithm, the VTK-m implementation has been extended to take advantage of all the hardware accelerators supported by VTK-m.