Alpine/ZFP

Computational science applications generate massive amounts of data from which scientists need to extract information and visualize the results. Performing the visualization and analysis tasks in situ, while the simulation is running, can lead to improved use of computational resources and reduce the time the scientists must wait for their results. The ALPINE/ZFP project is delivering in situ visualization and analysis infrastructure and algorithms including a data compression capability for floating-point arrays to reduce memory, communication, I/O, and offline storage costs.

Many high-performance simulation codes write data to disk to visualize and analyze it after the simulation is completed. Given the exascale I/O bandwidth constraints, this process will need to be performed in situ to fully utilize the exascale resources. In situ data analysis and visualization selects, analyzes, reduces, and generates extracts from scientific simulation while the simulation is running to overcome bandwidth and storage bottlenecks associated with writing the full simulation results to the file system. The ALPINE/ZFP project produces in situ visualization and analysis infrastructure that will be used by the exascale applications along with a lossy compression capability for floating point arrays.

The ALPINE development effort focuses on delivering exascale visualization and analysis algorithms that will be critical for exascale applications; developing an exascale-capable infrastructure for in situ algorithms and deploying it into existing applications, libraries, and tools; and engaging with exascale application teams to integrate ALPINE with their software. This capability will leverage existing, successful software, ParaView and VisIt, including their in situ libraries Catalyst and Libsim, by integrating and augmenting them with ALPINE capabilities to address the challenges of exascale.

Overcoming the performance cost of data movement is also critical. With deepening memory hierarchies and dwindling per-core memory bandwidth due to increasing parallelism, even on-node data motion makes for a significant performance bottleneck and primary source of power consumption. The ZFP software is a floating-point array primitive that mitigates this problem using very high-speed, lossy (but optionally error-bounded) compression to significantly reduce data volumes and I/O times. The ZFP development effort focuses on extending ZFP to make it more readily usable in an exascale computing setting by (1) parallelizing it on both CPU and GPU, while ensuring thread safety; (2) providing bindings for multiple programming languages; (3) adding new functionality; (4) hardening the software and adopting best practices for software development; and (5) integrating ZFP with a variety of exascale applications, I/O libraries, and software tools.

PI: Jim Ahrens, Los Alamos National Laboratory

Collaborators: Los Alamos National Laboratory, Lawrence Livermore National Laboratory, University of Oregon, Kitware, Inc.

Progress to date

- The ALPINE/ZFP team completed a layer on top of VTK-m for ALPINE algorithms, fully integrated the ALPINE infrastructure into ParaView and VisIt to support ALPINE algorithms to run both on CPUs and GPU accelerators, added ParaView visualization support, and made a production release of the code.
- The team demonstrated parallel implementations of core algorithms and automatic data selection methods in ALPINE, including feature-centric analysis, topolocation analysis, adaptive sampling, and Lagrangian analysis.
- The team developed OpenMP and CUDA parallel compression and decompression in ZFP that support up to 150 GB/s throughput, thus accelerating data transfer between CPU and GPU, extended compressed-array classes to be thread safe, and extended ZFP to support lossless compression and data preconditioning to improve compression of unstructured data.