My primary research interests lie in parallel I/O middleware, file systems and distributed systems. With the upcoming deployment of Exascale computers, a generic and intelligent high-level I/O interface are of pivotal importance for HPC scientific applications and the corresponding analysis and visualization tools. By identifying the existing problems with performance, scalability and usability of current I/O subsystems, my research efforts have been to provide software services for next generation HPC that can conveniently and efficiently interact with parallel storage systems such that computational scientists are not only able to delegate the I/O optimization to the high-level I/O interface during production cycle, but also quickly query and retrieve the data by using their analysis and visualization tools. The following sections outline my current research progress and future research directions.

Current Research

In the past two years, my research efforts have been dedicated on improving write performance for HPC scientific applications. We have explored different I/O optimization techniques, such as buffering and asynchronous function call for independent I/O operations. We have also developed various scheduling algorithms to improve average response time in collective I/O.

Adaptive I/O system: Before became a Ph.D. student, I worked at National Center of Computational Science (NCCS) group at Oak Ridge National Laboratory (ORNL) and participated in developing the adaptive I/O system (ADIOS) on Jaguar, JaguarPF, and Blue Gene/P machines there. This I/O system provides easy-to-use APIs, which allows computational scientists independently select different I/O methods for individual variables such that a set of best performed methods can be applied within one application based on I/O patterns and underlying hardware. For the GTC fusion code which scaled up to 16384 cores by then, [1], [2] and [3] showed the write bandwidth by using their self-defined intermediate file format could be comparable with the underlying file system's peak capability.

Data model in PnetCDF: In order for PnetCDF I/O library to support a broad range of data models, we proposed a new scheme for representing these data models by using graph transformation where each variable is represented as a vertex while the edge labels the dependency between its two end points. The metadata and data representation is designed in such as way that it conforms to the existing PnetCDF formats. A collection of APIs have also been implemented to abstract the operations on these graph representation. The work has been applied to Adaptive Mesh Refinement(AMR) applications, such as Flash code and Chombo I/O benchmark and we have shown the striking improvement regarding checkpointing's write performance[4].

I/O requests scheduling: In collective I/O, MPI processes exchange requests so that the rearranged requests can result in the shortest file system access time. Scheduling the exchange sequence determines the response time of participating processes. Existing implementations that simply follow the increasing order of file offsets do not necessary produce the best performance. To minimize the average response time, we propose three scheduling algorithms that consider the number of processes per file stripe and the number of accesses per process. Our experimental results demonstrate improvements of up to 50% response time using two synthetic benchmarks and a high-resolution climate application.

Future Agenda

As we move forward through Petascale to Exascale computing, the opportunities for groundbreaking science will continue to grow, yet the complexity of gaining insight will also propor-

tionally increase. One of the challenges is the difficulties of performing high performance I/O for both simulations and analysis. As a large amount of efforts have been dedicated on the data producing phase, the data retrieval part becomes even more critical in this big-data era.

Parallel Programming Model: Timely and cost-effective processing of large datasets has become a critical ingredient for the success of many academic, government, and industrial organizations. The combination of MapReduce framework and cloud computing is an attractive proposition for these organizations. Despite the success of the existing distributed platforms and programming models, such as MapReduce and Dryad, their limitation on iterative algorithms strive us to develop alternative mechanisms to gear for more generalized machine learning algorithms. Currently I am interning in the Extreme computing group at Microsoft Research and my research is focused on designing and developing a new programming model better suited for the iterative algorithms, such as PageRank, Gaussian descent algorithms, k-means and other machine learning algorithms. Along with the Windows' Azure platform, this will be beneficial for computational scientists to different iterative machine learning algorithms to analyze their large volume of data and don't need worry about the system performance, scalability and fault tolerance.

Unified Data Model: Scientific applications exhibit distinct grid types and computation and communication patterns, such as structured AMR, unstructured finite element and spectral elements. From a storage and I/O perspective, these applications exhibit distinct data organization and access pattern during simulation, analysis and visualization. However, these codes continue to interact with I/O data libraries much the same as they have since 1990s. The mismatch of the data models between scientific application and I/O libraries is growing as applications go to more complex data model and interactions. This mismatch is making it more difficult to achieve I/O performance close to system peak capabilities, while also not supporting the full range capabilities in today's computational scientific data models. In order for HPC applications to interact conveniently and efficiently with storage through abstractions that match their data models, I would like to focus my future research on (1) constructing a unified, high-level data model that maps naturally onto a set of data model motifs used in a representative set of HPC scientific applications; (2) developing a data model storage library that supports the unified data model, provides efficient storage data layouts, incorporates optimizations to enable exascale operation, and is resilient to failures; (3) assessing the performance of this approach through the construction of new I/O benchmarks to the use of existing I/O benchmarks for each of the data model motifs.

References

[1] C. Jin, S. Klasky, J. Lofstead and et al, "Adaptive IO System", *Cray User Group Conference*, Helsinki, May 2008.

[2] J. Lofstead, S. Klasky, K. Schwan and C. Jin, "Flexible IO and Integration for Scientific Codes Through The Adaptable IO System (ADIOS)", *ACM/IEEE International Symposium on High Performance Distributed Computing*, Boston, June 2008

[3] Chapter "High Throughput Data Movement" in "Scientific Data Management: Challenges, Existing Technologies, and Deployment", Chapman and Hall, ISBN 9781420069808, 2009

[4] Kui Gao, C. Jin, "Supporting Computational Data Model Representation with High-performance I/O in Parallel netCDF", IEEE International Conference on High Performance Computing (HiPC), 2011

[5] C. Jin et al, "Improving the Average Response Time in Collective I/O", *18th European MPI Users' Group Meeting*, EuroMPI, Santorini, Greece, September 15-18, 2011.