DOE Award No. DE-FG02-07ER25747

A Lightweight, High Performance I/O Management Package for Data-Intensive Computing

Progress Report (3/2008 – 2/2011)

U.S. Department of Energy

Office of Science

June 2011

Jun Wang

Department of Electrical Engineering and Computer Science,

University of Central Florida

Phone: (407) 823-0449 Email: jwang@eecs.ucf.edu

Project Website
http://www.eecs.ucf.edu/~jwang/research-lightIO.html

Background and Computing Infrastructure

The PI transferred from University of Nebraska Lincoln (UNL) to University of Central Florida (UCF) in Summer 2006. His research team in Computer Architecture and Storage System (CASS) laboratory has been working at the same project collaboratively with DOE computer scientists Drs. Robert Ross and Rajeeve Thakur at Argonne National Lab (ANL) in the new institution. Recently, we have been working with Gary Grider, James Nunez, John Bent, Meghan Wingate and Salman Habib at Los Alamos National Lab (LANL) to further some of the works.

We have set up a 16-node Dell Power-edge 1950 server cluster named CASS for in-house prototyping with different parallel file systems. It was configured with a dual dual-core Intel Xeon processor, 4 GB SDRAM and two 144 GB SAS hard drives per node and interconnected via a 48-port 1Gbps Ethernet switch. Recently, we scaled up this onsite cluster to 48-node by adding a Sun Solaris cluster.

We received an NERSC Allocation Year 2008 Startup award in AY 2008 that runs from January 8, 2008 through January 12, 2009. We renewed it for Year 2011. We have experimented with several NERSC super computers such as Franklin to analyze the real-world file system metadata characteristics in Lustre file system. More recently, we have been closely working with computer scientists from Los Alamos National Laboratory on extending and evaluating our prototypes.

PI and student investigators

The CASS group consists of five Ph.D. students at the moment. In all, about seven Ph.D. graduate students have been partially sponsored by the grant and working on the project together with the PI. Two successfully defended their Ph.D. dissertation in summer 2008 and summer 2010.

- Mr. Peng Gu, defended his Ph.D. dissertation work titled “Metadata and Data Management in High-performance File and Storage Systems” in July 2008
- Ms. Saba Sehrish, defended her PhD dissertation titled “Improving Performance and Programmer Productivity for I/O-Intensive High Performance Computing Applications” in July 2010.
- Mr. Grant Mackey, a 3rd year Ph.D. student (held a joint appointment with Los Alamos National Lab)
- Mr. Christopher Mitchell, a 3rd year Ph.D. student (held a joint appointment with Los Alamos National Lab)
- Mr. Junyao Zhang, a 2nd year Ph.D. student. He defended his Master thesis on “Research on Reverse Lookup Problem in Distributed File System” in Fall 2010.
- Ms. Lu Cheng, a first year Ph.D. student. She defended her Master thesis on “Concentric layout, a new data layout for matrix set in Hadoop file system” in Fall 2010.
- Ms. Zhengkai Wu, a first year Ph.D. student. She received her Master degree in Spring 2011.

Summary of Progress of Scientific and Technical Work

Our group has been working with ANL collaborators on the topic “bridging the gap between parallel file system and local file system” during the course of this project period. We visited Argonne National Lab -- Dr. Robert Ross’s group for one week in the past summer 2007. We looked over our current project progress and planned the activities for the incoming years 2008-09. The PI met Dr. Robert Ross several times such as HEC FSIO workshop 08, SC’08 and

SC’10. We explored the opportunities to develop a production system by leveraging our current prototype to (SOGP+PVFS) a new PVFS version. We delivered SOGP+PVFS codes to ANL PVFS2 group in 2008.We also talked about exploring a potential project on developing new parallel programming models and runtime systems for data-intensive scalable computing (DISC). The methodology is to evolve MPI towards DISC by incorporating some functions of Google MapReduce parallel programming model. More recently, we are together exploring how to leverage existing works to perform (1) coordination/aggregation of local I/O operations prior to movement over the WAN, (2) efficient bulk data movement over the WAN, (3) latency hiding techniques for latency-intensive operations.

Since 2009, we start applying Hadoop/MapReduce to some HEC applications with LANL scientists John Bent and Salman Habib [1]. Another on-going work is to improve checkpoint performance at I/O forwarding Layer for the Road Runner super computer with James Nuetz and Gary Gridder at LANL. Two senior undergraduates from our research group did summer internships about high-performance file and storage system projects in LANL since 2008 for consecutive three years. Both of them are now pursuing Ph.D. degree in our group and will be 4th year in the PhD program in Fall 2011 and go to LANL to advance two above-mentioned works during this winter break. Since 2009, we have been collaborating with several computer scientists (Gary Grider, John bent, Parks Fields, James Nunez, Hsing-Bung Chen, etc) from HPC5 and James Ahrens from Advanced Computing Laboratory in Los Alamos National Laboratory. We hold a weekly conference and/or video meeting on advancing works at two fronts: the hardware/software infrastructure of building large-scale data intensive cluster and research publications. Our group members assist in constructing several onsite LANL data intensive clusters. Two parties have been developing software codes and research papers together using both sides’ resources.

Next, we summarize each task in detail.

Task I Segment-structured On-disk data Grouping and Prefetching (SOGP) for parallel file systems

We developed Segment-structured On-disk data Grouping and Prefetching (SOGP), a technique that leverages additional local storage to boost the local data read performance for parallel file systems, especially for those applications with partially overlapped access patterns [2,3].

Recent years have seen growing research activities in various parallel file systems, such as Lustre, IBM's GPFS, Ceph, the Panasas PanFS File System, and PVFS. In many cases, parallel file systems use a local file system or object store to serve as a data reservoir. Because of the interfaces used to access these local resources, local storage systems are unaware of the behavior of high-level parallel applications that talk directly to parallel file systems. Likewise, the parallel file system itself is not aware of the underlying local storage organization and operation for the same reason. In other words, there is an information gap between local file system and parallel file system, and as a result access locality from applications often gets lost. Specifically, the local storage system can only see accesses to separate pieces of large parallel files. Emerging Object Storage Device (OSD) interfaces, used by parallel file systems, do appear to present a more appropriate interface for local storage resources, but at this time the OSD interface does not address this knowledge gap.

We note that *partially overlapped accesses* are becoming more common in many emerging scientific and engineering applications: these applications access the same data regions more than once during their execution. Although similar noncontiguous patterns were reported in HPC community one decade ago, parallel file system and local file system architectures have not been focused on addressing these patterns. Examples of this pattern of access are common in mapping, Geographic Information Systems (GIS), and visualization applications.

The main ideas behind SOGP are to store a copy of data that is often accessed in a more efficient organization by grouping noncontiguous file I/O requests and storing these groups (called *segments*) on a local disk partition, and to use this more efficient organization to improve the performance of prefetching at the local storage level, better catering to the needs of parallel file system. We conducted comprehensive experiments for performance evaluation using our local 16-node server cluster and ANL Chiba-City 512-node cluster. We replayed several scientific and engineering applications and benchmarks as diverse as Flash I/O, mpi-BLAST, non-contig, mpi-tile-io, and IOR to gauge the effectiveness. We concluded that our SOGP-enhanced PVFS scheme outperforms an EXT3-based PVFS by 39% to 230% in terms of aggregate I/O bandwidth in a testbed cluster system. In 2007, we visited Dr. Rob Ross's team at Argonne National Lab and delivered a high-performance local storage package with 2000+ lines of c codes.

In the last several months, we further gauged the effectiveness of SOGP for two real-world scientific applications, CP2K and SIESTA respectively [3]. CP2K is a freely available program under GPL licence, written in Fortran 95, to perform atomistic and molecular simulations of solid state, liquid, molecular and biological systems. SIESTA (Spanish Initiative for Electronic Simulations with Thousands of Atoms) is both a method and a computational implementation, to perform electronic structure calculations and ab initio MD simulations of molecules and solids.

We performed two kinds of test for CP2K on CASS cluster, one for fixed problem size and one of scaled problem size. First, given a fixed problem size, we varied the shape and size of the simulation cell in a three-dimension Cartesian space and performed four combinations tests. In general, CP2K shows good scaling performance in all experiments. For the cubic case with 60 clients, CP2K on PVFS/Ext3 presents a speedup of less than 30, showing a scaling efficiency of less than 50%. However, for the same experiment, CP2K on PVFS/SOGP achieved a speedup of 40, showing a scaling efficiency of around 66%. The final performance difference based on these two different platforms is as significant as 47%. Second, when studying a scaled problem case, we increased the number of clients that would result in larger aggregated data set being accessed. The performance gain we obtain via PVFS/SOGP over PVFS/Ext3 ranges from 17% (in the single client case) to 94% (in the 60 client case).

We also examined the time taken for one Self Consistent Field (SCF) iteration of the SIESTA application. This includes a matrix diagonalization and some numerical real space grid integrations. There are several modes of parallelism in SIESTA but the one tested was the most communications intensive. Given one client, the PVFS/SOGP based system was marginally faster than PVFS/Ext3 for the Siesta benchmark. However, under a 12-client case, PVFS/SOGP runs 1.56 times faster than PVFS/Ext3, and 2.16 times faster increased to a 24-client case. In this instance benchmarks were performed up to 60 processors.

For the next year, we will first try more real-world applications to further gauge SOGP effectiveness and scalability. Second, we plan to port SOGP to other parallel file systems such as Lustre and GPFS to further gauge its portability.

Task II Metadata prefetching in parallel file systems

We investigated the fundamental differences between data and metadata access patterns, which is a point largely neglected by previous prefetching algorithms and thus laid out the foundation for developing new metadata prefetching algorithms [5,9].

To find out the difference between file data and metadata size distribution, we studied the files stored on Franklin supercomputer in last few months [9]. Franklin is a massively parallel processing (MPP) system with 9,660 compute nodes, serving more than 1,000 users at National Energy Research Scientific Computing Center and employing an Lustre file system. The collection of file size distribution is somewhat straight forward compared with the metadata size case. We simply run "ls -lR /" command on the head node and then use a script to filter out the file size information from the output. In this study, we collected the size information for 8,209,710 regular files and 612,248 directories out of a 350 TB space. Based on our file data and metadata size distribution research, we observe that compared with typical file size, metadata are relatively small. We project that the same conclusion holds for petabyte scale storage system if there is no significant change on the way the file systems manage their data and metadata. Consequently, in order to achieve optimal performance, a new prefetching algorithm that considers the size differences between data and metadata is clearly desirable.

We developed Nexus, a novel weighted-graph-based prefetching algorithm specifically designed for clustered metadata servers. Aiming at the emerging MDS-cluster-based storage system architecture and exploiting the characteristic of metadata access, our prefetching algorithm distinguishes itself in the following aspects. First, Nexus exploits the ability to look ahead farther than the immediate successor to make wiser predictions. Sensitivity study shows that the best performance gain is achieved when the look-ahead history window size is set to 5. Second, based on the wiser prediction decision, aggressive prefetching is adopted in our Nexus prefetching algorithm to take advantage of the relatively small metadata size. Our study shows that prefetching 2 as a group upon each cache miss is optimal under the two particular traces studied. Conservative prefetching lose the chance to maximize the advantage of prefetching, and too aggressive but not so accurate prefetching might hurt the overall performance by introducing extra burden to the disk and polluting the cache. The relationship strengths of the successors are differentiated in our relationship graph by assigning variant edge weights. Four approaches for edge weight assignment were studied in our sensitivity study. The results show that the linear decremental assignment approach represents the most accurate strength for the relationships. Third, in addition to server-oriented grouping, we also explored client-oriented grouping as a way to capture better metadata access locality by differentiating between the sources of the metadata requests. Sensitivity study results show the latter approach’s consistent performance gain over the former approach, confirming our assumption. Other than focusing on the prefetching accuracy — an indirect performance measurement, we pay our attentions to the more direct performance goal — cache hit rate improvement and average response time reduction. Simulation results show remarkable performance gains on both hit rate and average response time over conventional and state of the art caching/prefetching algorithms.

Task III Scalable metadata lookup

We have investigated how to use Bloom Filter technique to assist in load balancing in a large-scale cluster based parallel file system that employs a split data and metadata architecture. We develop a Bloom filter based file lookup scheme such as Bloom-filter-array (HBA) [5, 12, 10]. We implemented the HBA prototype on the Linux kernel 2.4.21 as an I/O daemon running on each metadata server. All internal communications use TCP/IP and the request forwarding between metadata servers is implemented by using IP-IP encapsulation. We provide a C library that includes functions analogous to the UNIX/POSIX functions, such as HBA open, HBA close, and HBA stat. We run experiments on a Sandhills cluster that has 40 nodes, each equipped with dual-AMD processors. The experimental results based on our prototype implementation indicate that the HBA scheme maintains a strong scalability in increasing the aggregate throughput and reducing the latency of metadata operations.

Task IV Lock technique in MPI/IO and PVFS

Many scientific applications require high performance concurrent IO accesses to a file by multiple processes. Those applications rely indirectly on atomic IO capabilities in order to perform updates to structured datasets, such as those stored in HDF5 format files. Current support for atomicity in MPI-IO is provided by locking around the operations, imposing lock overhead in all situations, even though in many cases these operations are non-overlapping in the file. We propose to isolate non-overlapping accesses from overlapping ones in independent I/O cases, allowing the non-overlapping ones to proceed without imposing lock overhead. To enable this we have implemented an efficient conflict detection algorithm in MPI-IO using MPI file views and datatypes. Our results show that conflict detection algorithm incurs a minimal overhead, and perform within 3.6% of the ideal case (i.e. concurrent access with no-locks) [6, 17]. When the file view for a process does not overlap with the file views of other processes, locking is not required; there will be no conflicts. Our approach reduces the locking overhead at minimum for the non-overlapping regions but handles the overlapped regions using the locks.

Task V DCA at iSCSI storage server

With the emergence of data intensive applications, recent years have seen a fast growing volume of I/O traffic propagated through the local I/O interconnect bus. This raises up a question for storage servers on how to resolve such a potential bottleneck. We present a hierarchical Data Cache Architecture called DCA to effectively slash local interconnect traffic and thus boost the storage server performance. A popular iSCSI storage server architecture is chosen as an example. DCA is composed of a read cache in NIC called NIC cache and a read/write unified cache in host memory called Helper cache. NIC cache services most portions of read requests without fetching data via PCI bus, while Helper cache 1) supplies some portions of read requests per partial NIC cache hit; 2) directs cache placement for NIC cache and 3) absorbs most transient writes locally. We develop a novel State Locality Aware cache Placement algorithm called SLAP to improve NIC cache hit ratio for mixed read and write workloads. To demonstrate the effectiveness of DCA, we develop a DCA prototype system and evaluate it with an open source iSCSI implementation under representative storage server workloads. Experimental results showed that DCA can boost iSCSI storage server throughput by up to 121% and reduce the PCI traffic by up to 74% compared with an iSCSI target without DCA [4].

Task VI Parallel programming models for data-intensive scalable computing and others

We present a case for automating the selection of MPI-IO performance optimizations, with an ultimate goal to relieve the application programmer from these details, thereby improving their productivity. Programmer productivity has always been overlooked as compared to the performance optimizations in high performance computing community. Recently, we develop RFSA, a Reduced Function Set Abstraction based on an existing parallel programming interface (MPI-IO) for I/O [7]. MPI-IO provides high performance I/O function calls to the scientists/engineers writing parallel programs; who are required to use the most appropriate optimization of a specific function, hence limits the programmer productivity. Therefore, we propose a set of reduced functions with an automatic selection algorithm to decide what specific MPI-IO function to use. We implement a selection algorithm for I/O functions like read, write, etc. RFSA replaces six different flavors of read and write functions by one read and write function. By running different parallel I/O benchmarks on both medium-scale clusters and NERSC supercomputers, we show that RFSA functions impose minimal performance penalties.

Additionally, we examine other research problems about block distribution policy in large-scale replication storage systems [8], energy-efficient high-performance storage systems [11, 13] and task scheduling techniques in data-intensive scalable computing [14].

Task VII Collaborative works with Los Alamos National Laboratory

We have been collaborating with several computer scientists (Gary Grider, John bent, Parks Fields, James Nunez, Hsing-Bung Chen, etc) from HPC5 and James Ahrens from Advanced Computing Laboratory in Los Alamos National Laboratory. We hold a weekly conference and/or video meeting on advancing works at two fronts: the hardware/software infrastructure of building large-scale data intensive cluster and research publications. Our group members assist in constructing several onsite LANL data intensive clusters. Two parties have been developing software codes and research papers (We published one work in IEEE IPDPS 2011 [18].) together using both sides resources on several specific on-going projects:

1) Compute and Data-Intensive Computing: Bridging the Gap.

Data-intensive computing is an emerging trend in computational science. Current HPC approaches to data-intensive computing result in excessive data migration between compute and storage resources. The time to migrate these large(TB/PB) datasets between resources by conventional means takes days to months. This overhead is unacceptable for applications which require data analysis after computation execution. We propose to remove the conventional HPC storage and replace it with a data intensive storage cluster. That is, when computationally intense jobs write data to permanent storage, they will now write to a data-intensive (network-attached) storage cluster rather than a traditional HPC file system. Utilizing a data-intensive storage resource removes the need for constant data migration between compute and storage resources to perform analysis on the output of computation-intensive workloads. Our comprehensive test on the QCD Lattice application suite shows a significant improvement in overall I/O time by using our proposed system over conventional HPC approaches.

2) Rethinking I/O in Large-Scale Visualization: Embracing Data Locality.

As simulation has gained footing as an equal to theory and experimentation, the demand for higher-resolution and longer-running models has pushed simulation computing requirements progressively higher. These progressively higher requirements have pushed computational capability through the PetaFLOP barrier while also demanding the storage of Petabyes of information. All of this raw simulation output does nothing to answer the original scientific questions posed without analysis and visualization of the results. The visualization process, however, is very I/O intensive which leads to long delays in seeing the results of these simulation runs. Delays that are rooted in the inability to retrieve data fast enough by the visualization pipeline's readers as well as due to machine failure which requires a complete restart of the process. In response to these issues, we propose a revamped reader design which relies on the Hadoop Distributed File System (HDFS) to provide data locally to the individual pipeline nodes. In achieving data locality to the visualization pipeline processes we are effectively able to ship the computation to the data rather than the, more expensive, other way around. In addition, we propose a rethinking on the failure mechanism of the MPI-parallelized visualization system such that node loss does not cause a lengthy restart process nor jeopardize data locality to the restarted process. This new mechanism gracefully restarts the server processes in the event of node failure and reassigns the lost node's workload to an in-tact node that houses a replica of the original node's data. Full implementation of these ideas was completed on the widely adopted ParaView visualization package and tested against the current release build of the package. Testing against a high end NFS server and a commonly deployed Parallel File System showed our implementation leveraging HDFS performed better respectively in testing with real world simulation data sets on a large parallel visualization cluster.

Task VIII MRAP: A Novel MapReduce-based Framework to Support HPC Analytics Applications with Access Patterns

Due to the explosive growth in the size of scientific data sets, data-intensive computing is an emerging trend in computational science. Many application scientists are looking to integrate data-intensive computing into computational intensive High Performance Computing facilities, particularly for data analytics. We have observed several scientific applications which must migrate their data from an HPC storage system to a data-intensive one. There is a gap between the data semantics of HPC storage and data-intensive system, hence, once migrated, the data must be further refined and reorganized. This reorganization requires at least two complete scans through the data set and then at least one MapReduce program to prepare the data before analyzing it.

Running multiple MapReduce phases causes significant overhead for the application, in the form of excessive I/O operations. For every MapReduce application that must be run in order to complete the desired data analysis, a distributed read and write operation on the file system must be performed. Our contribution [15, 16] is to extend Map-Reduce to eliminate the multiple scans and also reduce the number of pre-processing MapReduce programs. We have added additional expressiveness to the MapReduce language to allow users to specify the logical semantics of their data such that 1) the data can be analyzed without running multiple data pre-processing MapReduce programs, and 2) the data can be simultaneously reorganized as it is migrated to the data-intensive file system. Using our augmented Map-Reduce system, MapReduce with Access Patterns (MRAP), we have demonstrated up to 33% throughput improvement in one real application, and up to 70% in an I/O kernel of another application.

Task X Concentric layout: a new scientific data distribution scheme in Hadoop file system

The data generated by scientific simulation, sensor, monitor or optical telescope has increased with dramatic speed. In order to analyze the raw data efficiently, data pre-process operation is needed to achieve better performance in data analysis phase. Current research shows an increasing tread of adopting Map-Reduce framework for large scale data processing. However, the data access patterns which generally applied to scientific data set are not supported by current MapReduce framework directly. The gap between the requirement from analytics application and the property of MapReduce framework motivates us to provide support for these data access pattern in Map-Reduce framework. In our work, we studied the data access patterns in matrix file and proposed concentric data layout to support matrix data access and analysis in Map-Reduce framework. Concentric data layout is a hierarchical data layout which maintains the dimensional property in data set. Contrary to the continuous data layout, concentric data layout stores the data within same sub-matrix into same chunk, and then stores chunks symmetrically. The concentric data layout organize the data before head, and optimize the afterward run of Map-Reduce application. The experiments indicate that the concentric data layout improves the overall performance, reduces the execution time by 38% when file size is 4GB, also it relieves the data overhead phenomenon and increases the useful data retrieve rate by 32% in average [16].

Publications, Software Deliverables, Reports and Talks

1. Introducing Map-Reduce to High-End Computing, with Grant Mackey, Julio Lopez (CMU), Saba Sehrish, John Bent (LANL), Salman Habib (LANL), Petascale Data Storage Workshop Supercomputing '08, November 17, 2008, Austin, Texas.
2. Bridging The Gap Between Parallel File Systems and Local File Systems: A Case Study with PVFS, with Peng Gu and Robert Ross (ANL). The 37th International conference on Parallel Processing 2008. September 8–12, Portland, Oregon, USA. A software package of 2000 lines of C codes has been integrated into a PVFS2 prototype.
3. Segment-Structured On-disk Grouping and Prefetching Algorithm for Parallel I/O Perfromance: A Case Study with PVFS, with Peng Gu, Robert Ross (ANL), Rajeev Thakur (ANL), to be submitted to IEEE transactions on parallel and distributed systems.
4. A New Hierarchical Data Cache Architecture for iSCSI Storage Server, with Christopher Mitchell, Xiaoyu Yao and Peng Gu. IEEE transactions on computers Vol. 58, No. 4, pp 1-15, April 2009.
5. HBA: Distributed Metadata Management for Large Cluster-based Storage Systems. Yifeng Zhu, Jiang Hong, Jun Wang, Xian Feng. IEEE transactions on Parallel and Distributed Systems. Vol. 19, No. 6, pp. 750-763, June 2008.
6. Saba Sehrish, Jun Wang, and Rajeev Thakur (ANL). Self-detecting Locks to Support MPI-IO Atomicity. EuroPVM/MPI September 2009.
7. Saba Sehrish, and Jun Wang. Smart Read/Write for MPI-IO. IPDPS'09: Proceedings of the 2009

IEEE International Symposium on Parallel and Distributed Processing. Pages: 1-8.
8. Shifted Declustering: An Ideal-placement Layout Scheme for Multi-way Replication Storage Architecture. Huijun Zhu, Peng Gu, and Jun Wang. The 22nd ACM International Conference on Supercomputing. June 7–12, 2008, Island of Kos, Aegean Sea, Greece.
9. Nexus: A Novel Weighted-Graph-Based Prefetching Algorithm for Metadata Servers in Petabyte Scale Storage Systems. Peng Gu, Jun Wang, Yifeng Zhu, Hong Jiang, IEEE transactions on Computers, Vol. 59, No. 1, pp1-15, January 2010.
10. Applications of Bloom Filters in Peer-to-peer Systems: Issues and Questions. Hailong Cai, Ping Ge, Jun Wang. The 2008 International Conference on Networking, Architecture, and Storages (NAS 2008).
11. Jun Wang, Xiaoyu Yao, and Huijun Zhu, Exploiting In-memory and On-disk Redundancy to Conserve Energy in Parity Disk Array. IEEE transactions on Computers. Vol. 57, No. 6, pp. 733-747, June 2008.
12. Hong Jiang, Yifeng Zhu, Jun Wang, and David Swanson, Scalable and Adaptive Metadata Management for High-end Computing, HEC FSIO R&D Workshop/HECURA FSIO PI Meeting '07, August 2007, Arlington, VA.
13. Jun Wang, Huijun Zhu, and Dong Li, eRAID: Conserving Energy in Conventional Disk based RAID System. IEEE Transactions on Computers. Vol. 57, No. 3, pp. 359-374, March 2008.
14. Saba Sehrish, PhD dissertation published in Summer 2010, "Improving Performance and Programmer Productivity for I/O-Intensive High Performance Computing Applications".
15. MRAP: A Novel MapReduce-based Framework to Support HPC Analytics Applications with Access Patterns, Saba Sehrish, Grant Mackey, Jun Wang and John Bent (LANL), the ACM High Performance Distributed Computing (ACM HPDC'10). June 2010, Chicago, IL, USA.
16. Lu Cheng, Pengju Shang, Saba Serish, Grant Mackey and Jun Wang, Concentric layout: a new scientific data distribution scheme in Hadoop file system, (2010). Proceedings of the 5th IEEE International Conference on Networking, Architecture, and Storage (NAS 2010). Macau SAR, China.
17. Saba Sehrish and Jun Wang., "Reduced Function Set Architecture for MPI-IO", Journal of Supercomputing, Accepted in 2010.
18. Christopher Mitchell, James Ahrens (LANL) and Jun Wang. VisIO: Enabling Interactive Visualization of Ultra-Scale, Time Series Data via High-Bandwidth Distributed I/O Systems. The 25th IEEE International Parallel & Distributed Processing Symposium May 16-20, 2011, Anchorage (Alaska) USA.

The publications and talks listed above are available upon request to the PI.