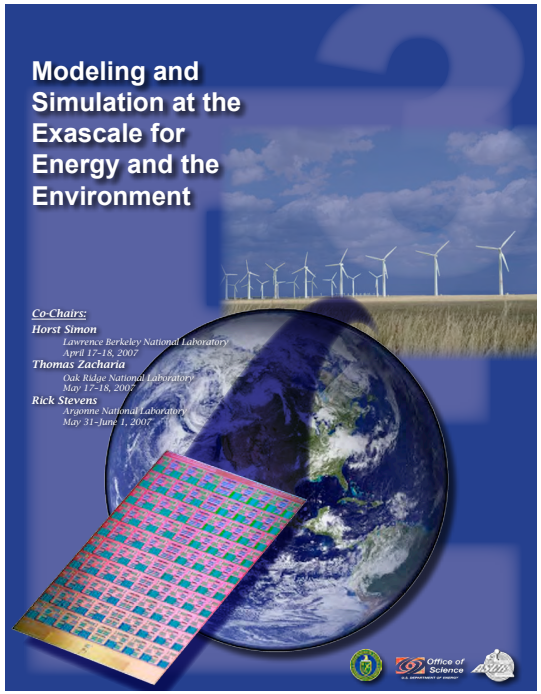
Storage in an Exascale World

Rob Ross
Mathematics and Computer Science Division
Argonne National Laboratory
rross@mcs.anl.gov

Exascale Computational Science

International effort to develop exascale compute systems by the end of the decade, enabling fundamental advances in many science domains.



- DOE Exascale Town Hall Meetings (2007)
- DOE Scientific Grand Challenges Workshops
  - Climate Science, Nov. 6-7, 2008
  - High Energy Physics, Dec. 9-11, 2008
  - Nuclear Physics, Jan. 26-28, 2009
  - …
  - Architectures and Technology, Dec. 8-10, 2009
  - Crosscutting Workshop, Feb. 2-4, 2010
- International Exascale Software Project
- DARPA/IPTO Extreme Scale Computing Study reports (2008-2009)

10^18 INTERNATIONAL EXASCALE SOFTWARE PROJECT

Exascale Systems: Potential Architecture

Systems	2009	2018	Difference
System Peak	2 Pflop/sec	1 Eflop/sec	O(1000)
Power	6 Mwatt	20 Mwatt	
System Memory	0.3 Pbytes	32-64 Pbytes	O(100)
Node Compute	125 Gflop/sec	1-15 Tflop/sec	O(10-100)
Node Memory BW	25 Gbytes/sec	2-4 Tbytes/sec	O(100)
Node Concurrency	12	O(1-10K)	O(100-1000)
Total Node Interconnect BW	3.5 Gbytes/sec	200-400 Gbytes/sec	O(100)
System Size (Nodes)	18,700	O(100,000-1M)	O(10-100)
Total Concurrency	225,000	O(1 billion)	O(10,000)
Storage	**15 Pbytes**	**500-1000 Pbytes**	**O(10-100)**
I/O	**0.2 Tbytes/sec**	**60 Tbytes/sec**	**O(100)**
MTTI	Days	O(1 day)	

From J. Dongarra, "Impact of Architecture and Technology for Extreme Scale on Software and Algorithm Design," Cross-cutting Technologies for Computing at the Exascale, February 2-5, 2010.

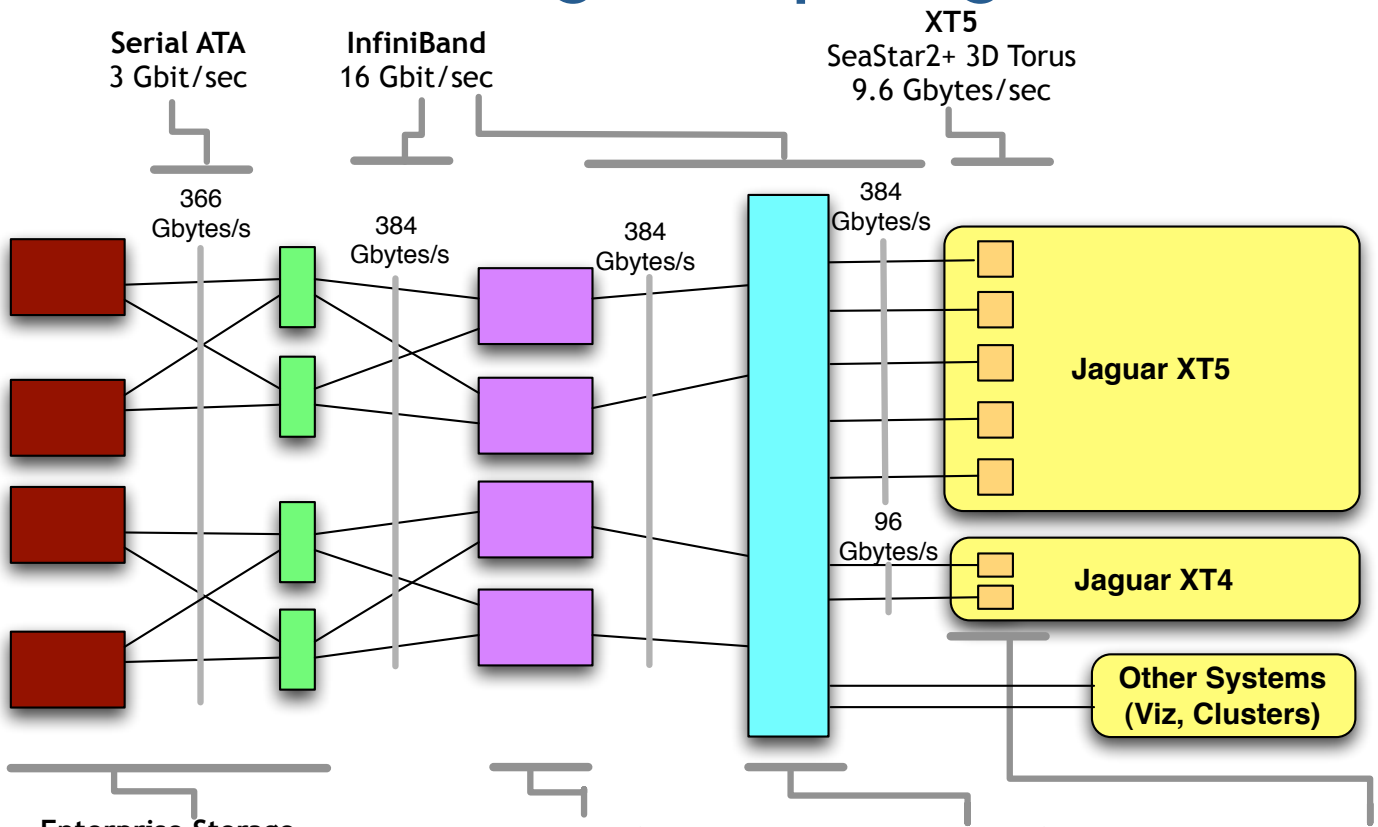
Outline

- The Present
  - Current Leadership storage systems
  - Argonne Blue Gene/P storage study
  - Argonne Blue Gene/P application I/O study
- The Exascale Era
  - Roles of I/O at Exascale
  - Performance disparity between I/O and compute
  - Research and development avenues to address gaps
- Conclusions

Outline

- **The Present**
  - Current Leadership storage systems
  - Argonne Blue Gene/P storage study
  - Argonne Blue Gene/P application I/O study
- The Exascale Era
  - Roles of I/O at Exascale
  - Performance disparity between I/O and compute
  - Research and development avenues to address gaps
- Conclusions

The Present: Oak Ridge Computing Platform



Enterprise Storage
controllers and large racks of disks are connected via InfiniBand.

48 DataDirect S2A9900 controller pairs with 1 Tbyte drives and 4 InifiniBand connections per pair

Storage Nodes
run parallel file system software and manage incoming FS traffic.

192 dual quad core Xeon servers with 16 Gbytes of RAM each

SION Network
provides connectivity between OLCF resources and primarily carries storage traffic.

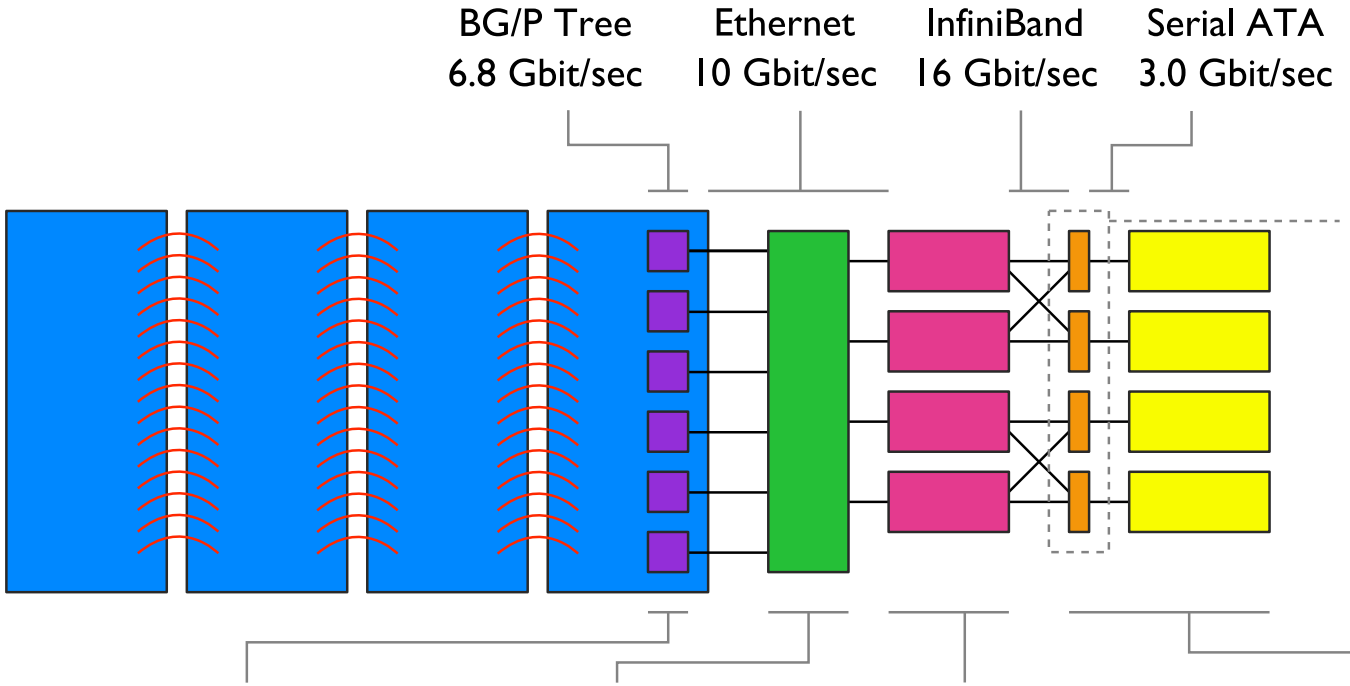
3000+ port 16 Gbit/sec InfiniBand switch complex

Lustre Router Nodes
run parallel file system client software and forward I/O operations from HPC clients.

192 (XT5) and 48 (XT4) one dual core Opteron nodes with 8 GB of RAM each

Figure compliments Galen Shipman (ORNL), Feb. 16, 2010.

The Present: Argonne Leadership Computing Facility



HW bottleneck is here. Controllers can manage only 4.6 Gbyte/sec.

Peak I/O system bandwidth is 78.2 Gbyte/sec.

Gateway nodes run parallel file system client software and forward I/O operations from HPC clients.

640 Quad core PowerPC 450 nodes with 2 Gbytes of RAM each

Commodity network primarily carries storage traffic.

900+ port 10 Gigabit Ethernet Myricom switch complex

Storage nodes run parallel file system software and manage incoming FS traffic from gateway nodes.

136 two dual core Opteron servers with 8 Gbytes of RAM each

Enterprise storage controllers and large racks of disks are connected via InfiniBand or Fibre Channel.

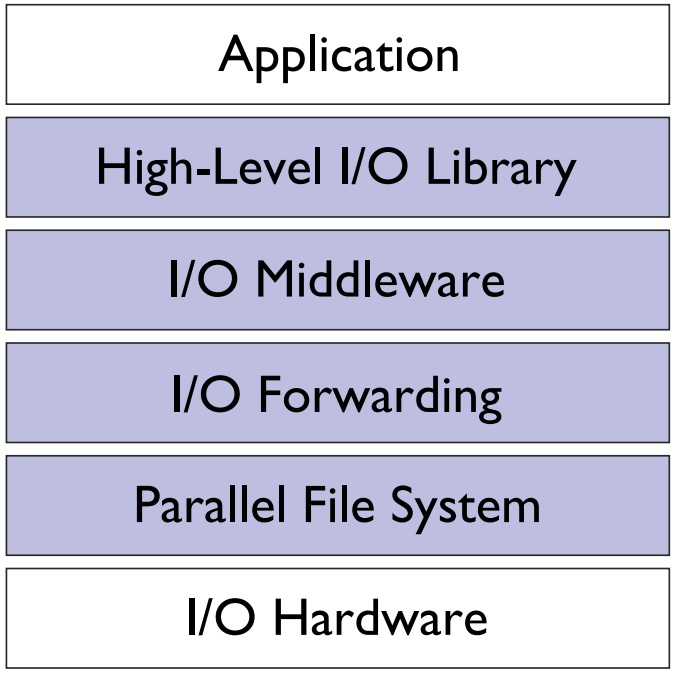
17 DataDirect S2A9900 controller pairs with 480 1 Tbyte drives and 8 InfiniBand ports per pair

Architectural diagram of the 557 TFlop IBM Blue Gene/P system at the Argonne Leadership Computing Facility.

Study of Argonne Blue Gene/P I/O System

- We recently performed an end-to-end study of I/O on the Intrepid system at Argonne, an IBM Blue Gene/P system.

- Goals:
  - Assess the limiting factors on performance for applications running at various scales
  - Assess the effectiveness of I/O system software in delivering performance for realistic application I/O patterns
  - Develop guidelines for improving performance on the system



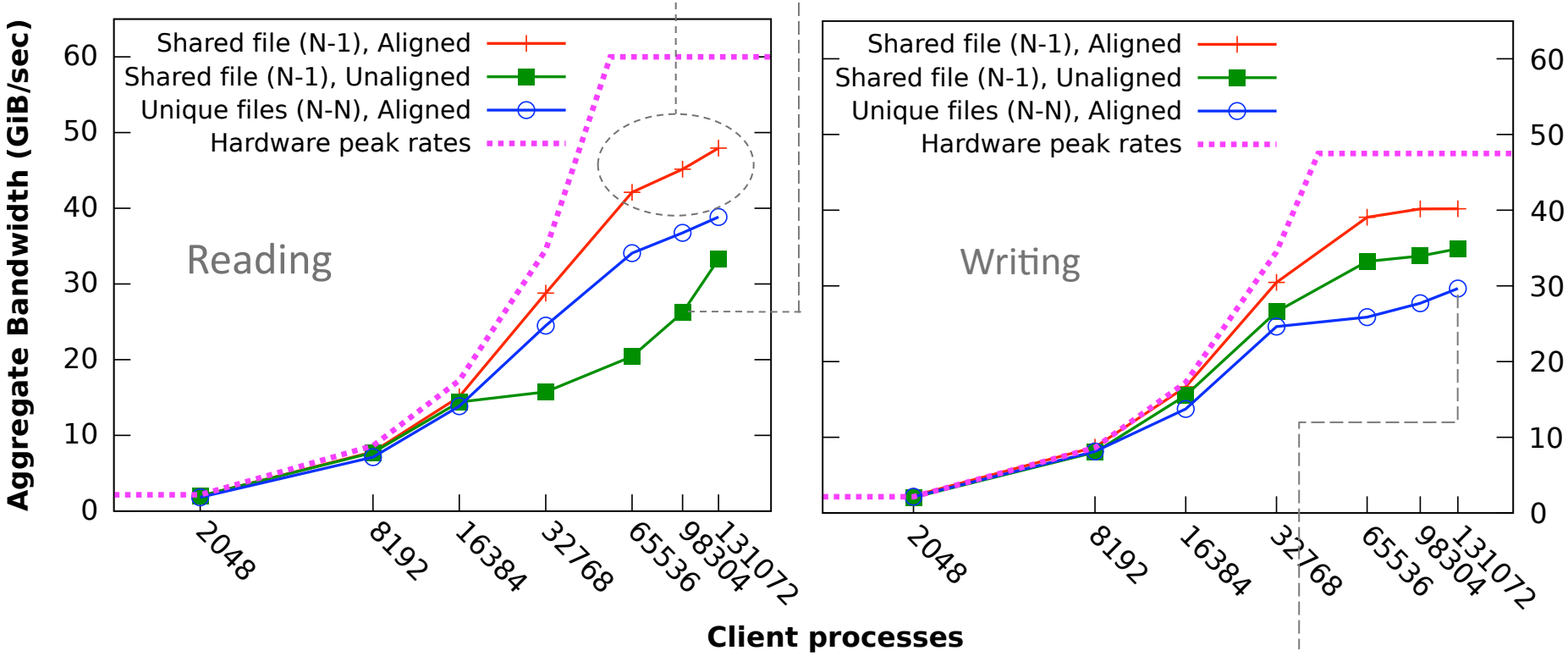
I/O software stack on the BG/P includes PVFS, the IBM ciod I/O forwarding system, ROMIO MPI-IO, and PnetCDF and HDF5.

S. Lang et al, "I/O Performance Challenges at Leadership Scale," in Proceedings of Supercomputing, November 2009.

End-to-End Scalability Tests (Using IOR)

Effective BW out of storage racks limited by scheduling and pipelining.

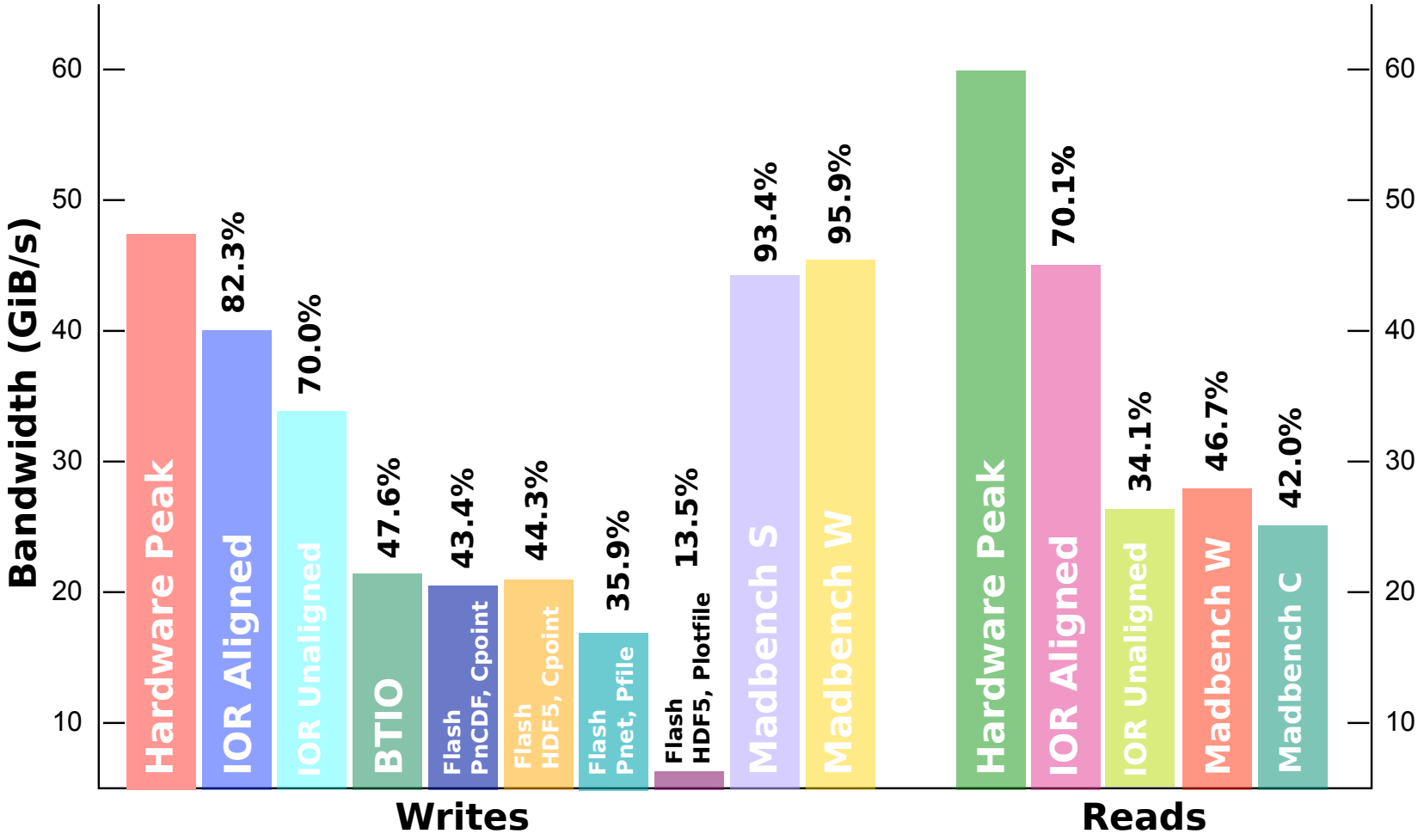
Unaligned requests are split, resulting in small (and odd-sized!) requests to servers



Unique file write allows hot spots in data placement, O(n) times more metadata updates for file size

I/O Benchmarks on ALCF Blue Gene/P

Synthetic and application I/O benchmarks often attain a significant fraction of peak, without tuning by I/O experts.



What We Learned from the I/O System Study

- Hardware is closely balanced, with workload characteristics defining the ultimate bottleneck for a given application.
- I/O forwarding software does not dramatically change how I/O systems behave for most workloads (with some exceptions).
- Application I/O benchmarks attain a significant fraction of the bandwidth seen by the best-behaved synthetic workloads (the Flash HDF5 plotfile case is explained in the paper).

- Based on this we would expect applications running on the system to see reasonable performance.

Characterizing Application I/O

How are are applications using the I/O system, and how successful are they at attaining high performance?

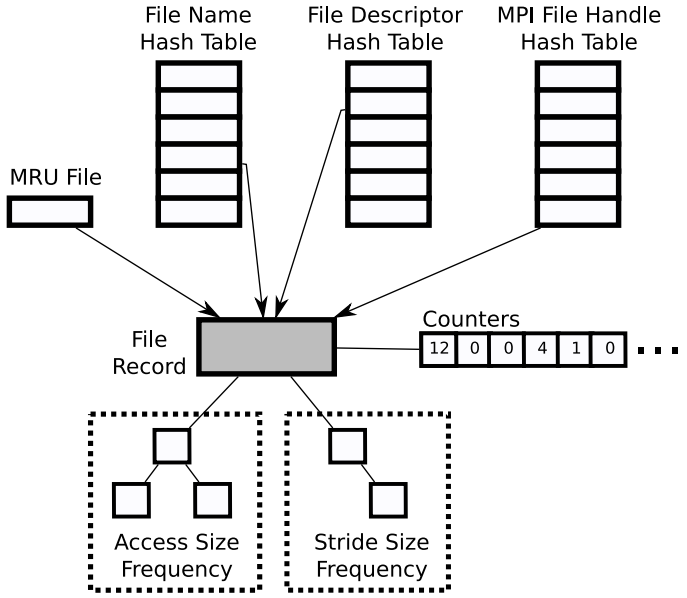
Darshan (Sanskrit for "sight") is a tool we developed for I/O characterization at extreme scale:

- No code changes, small and tunable memory footprint (~2MB default)
- Characterization data aggregated and compressed prior to writing
- Captures:
  - Counters for POSIX and MPI-IO operations
  - Counters for unaligned, sequential, consecutive, and strided access
  - Timing of opens, closes, first and last reads and writes
  - Cumulative data read and written
  - Histograms of access, stride, datatype, and extent sizes

http://www.mcs.anl.gov/darshan/
P. Carns et al, "24/7 Characterization of Petascale I/O Workloads," IASDS Workshop, held in conjunction with IEEE Cluster 2009, September 2009.

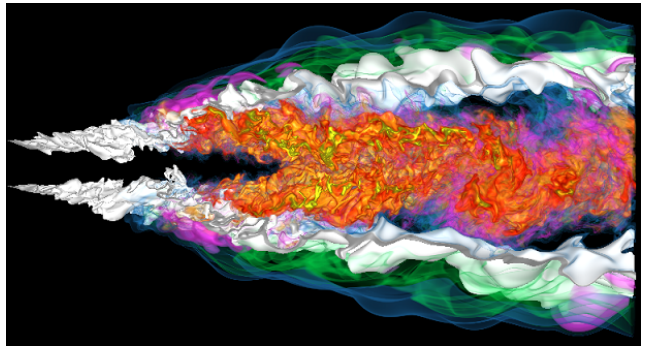
Darshan Internals

- Characterization centers around per-file records
  - Multiple hash tables allow relating accesses to one another
  - Falls back to aggregate (across files) mode if file limit is exceeded
- At output time, processes further reduce output size
  - Communicate to combine data on identical files accessed by all processes
  - Independently compress (gzip) remaining data
    - 32K processes writing a shared file leads to 203 bytes of compressed output
    - 32K processes writing a total of 262,144 files leads to 13.3MB of output

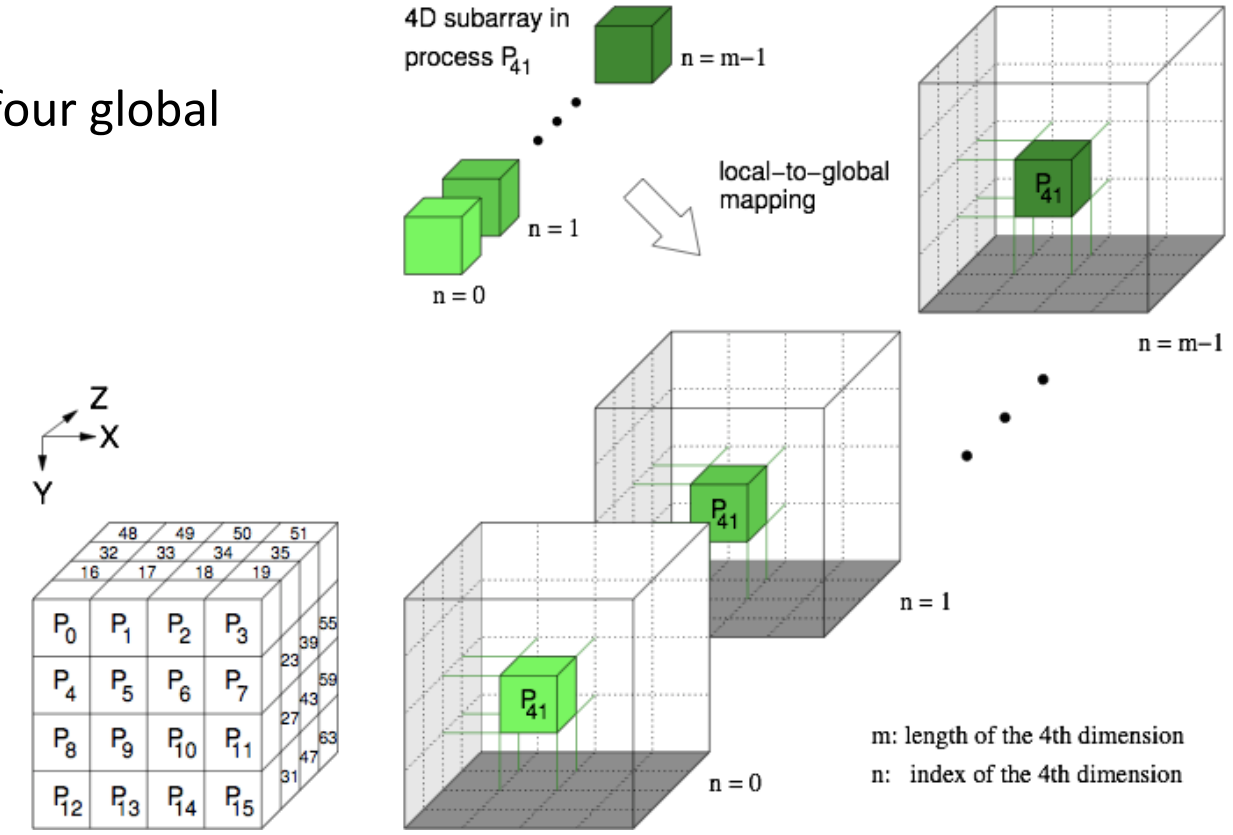


Multiple tables allow efficient location of file records by name, file descriptor, or MPI File handle.

S3D Turbulent Combustion Code



- S3D is a turbulent combustion application using a direct numerical simulation solver from Sandia National Laboratory
- Checkpoints consist of four global arrays
  - 2 3-dimensional
  - 2 4-dimensional
  - 50x50x50 fixed subarrays



Thanks to Jackie Chen (SNL), Ray Grout (SNL), and Wei-Keng Liao (NWU) for providing the S3D I/O benchmark, Wei-Keng Liao for providing this diagram, C. Wang, H. Yu, and K.-L. Ma of UC Davis for image.

Darshan, S3D, and I/O Optimizations

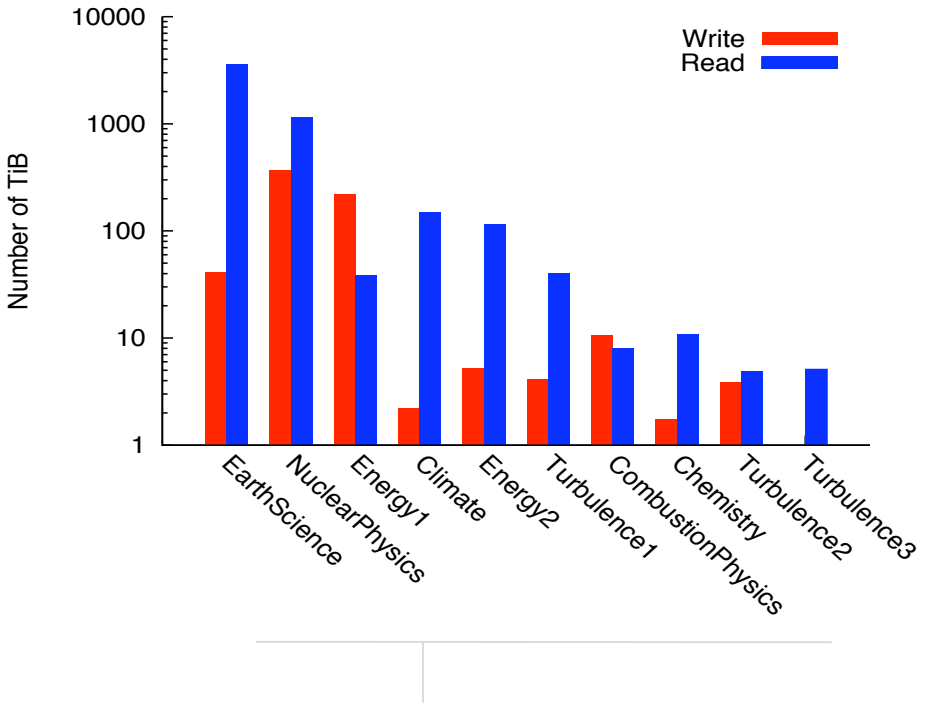
- Testing with PnetCDF output to single file, 16 processes
  - All MPI-IO optimizations disabled
  - Independent I/O optimization (data sieving) enabled
  - Collective I/O optimization (i.e., two-phase I/O) enabled

	No Optimizations	Data Sieving Enabled	Two-Phase Enabled
POSIX writes	102,401	81	**5**
POSIX reads	0	80	0
MPI-IO writes	64	64	64
Unaligned in file	102,399	80	4
Total written (MB)	6.25	**87.11**	6.25
Runtime (sec)	1443	11	6.0
Avg. MPI-IO time per proc (sec)	**1426.47**	4.82	0.60

Two Months of Application I/O on ALCF Blue Gene/P

- After additional testing and hardening, Darshan installed on Intrepid
- By default, all applications compiling with MPI compilers are instrumented

- Data captured from late January through late March of 2010
- Darshan captured data on 6,480 jobs (27%) from 39 projects (59%)
- Simultaneously captured data on servers related to storage utilization



Top 10 data producers and/or consumers shown. Surprisingly, most "big I/O" users read more data during simulations than they wrote.

P. Carns et al, "Storage Access Characteristics of Computational Science Applications," forthcoming.

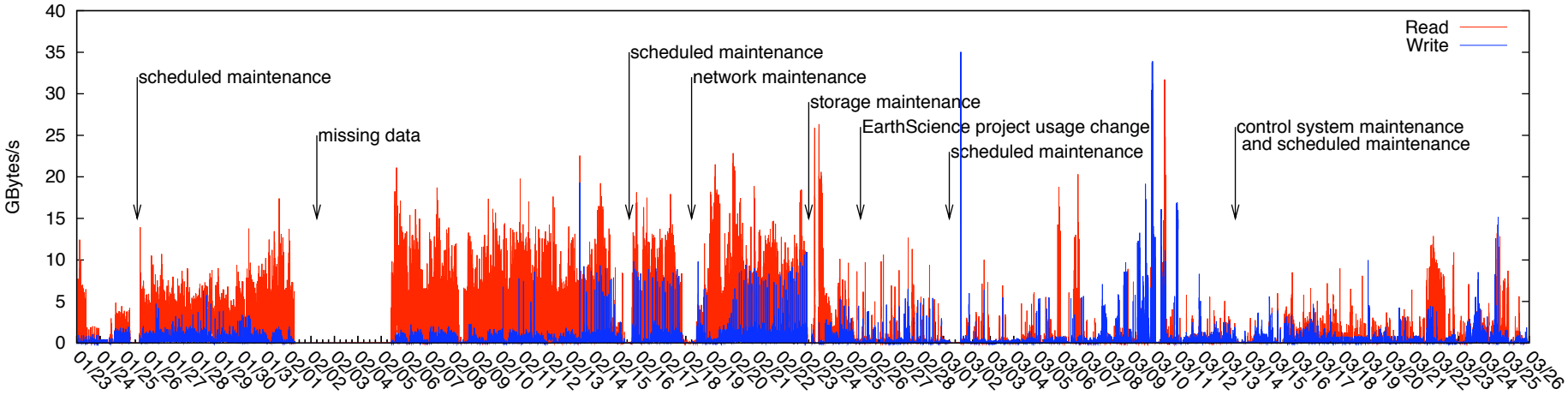
Real Application I/O on ALCF Blue Gene/P

Application	Mbytes/ sec/CN*	Cumulative MD	Files/ Proc	Creates/ Proc	Seq. I/O	Mbytes/ Proc
EarthScience	0.69	95%	140.67	98.87	65%	1779.48
NuclearPhysics	1.53	55%	1.72	0.63	100%	234.57
Energy1	0.77	31%	0.26	0.16	87%	66.35
Climate	0.31	82%	3.17	2.44	97%	1034.92
Energy2	0.44	3%	0.02	0.01	86%	24.49
Turbulence1	0.54	64%	0.26	0.13	77%	117.92
CombustionPhysics	1.34	67%	6.74	2.73	100%	657.37
Chemistry	0.86	21%	0.20	0.18	42%	321.36
Turbulence2	1.16	81%	0.53	0.03	67%	37.36
Turbulence3	0.58	1%	0.03	0.01	100%	40.40

* Synthetic I/O benchmarks (e.g., IOR) attain 3.93 - 5.75 Mbytes/sec/CN for modest job sizes, down to approximately 1.59 Mbytes/sec/CN for full-scale runs.

P. Carns et al, "Storage Access Characteristics of Computational Science Applications," forthcoming.

I/O on Today's Systems: A System Perspective



Aggregate I/O throughput on BG/P storage servers at one minute intervals.

- The I/O system is rarely **idle** at this granularity.
- The I/O system is also rarely at more than 33% of peak bandwidth.
- One particularly poor performing application can dramatically impact the system.

P. Carns et al, "Storage Access Characteristics of Computational Science Applications," forthcoming.

What We Learned in the Application I/O Study

- We need to keep giving I/O tutorials.
- Wide range of access patterns are seen, no one of which is obviously the most successful.
- High-level I/O libraries were rarely used in observed projects, but the applications don't seem to have achieved higher performance by avoiding them.
- I/O system is extremely underutilized.

- We are now beginning to work with our top I/O users to assist them in better using the storage system.

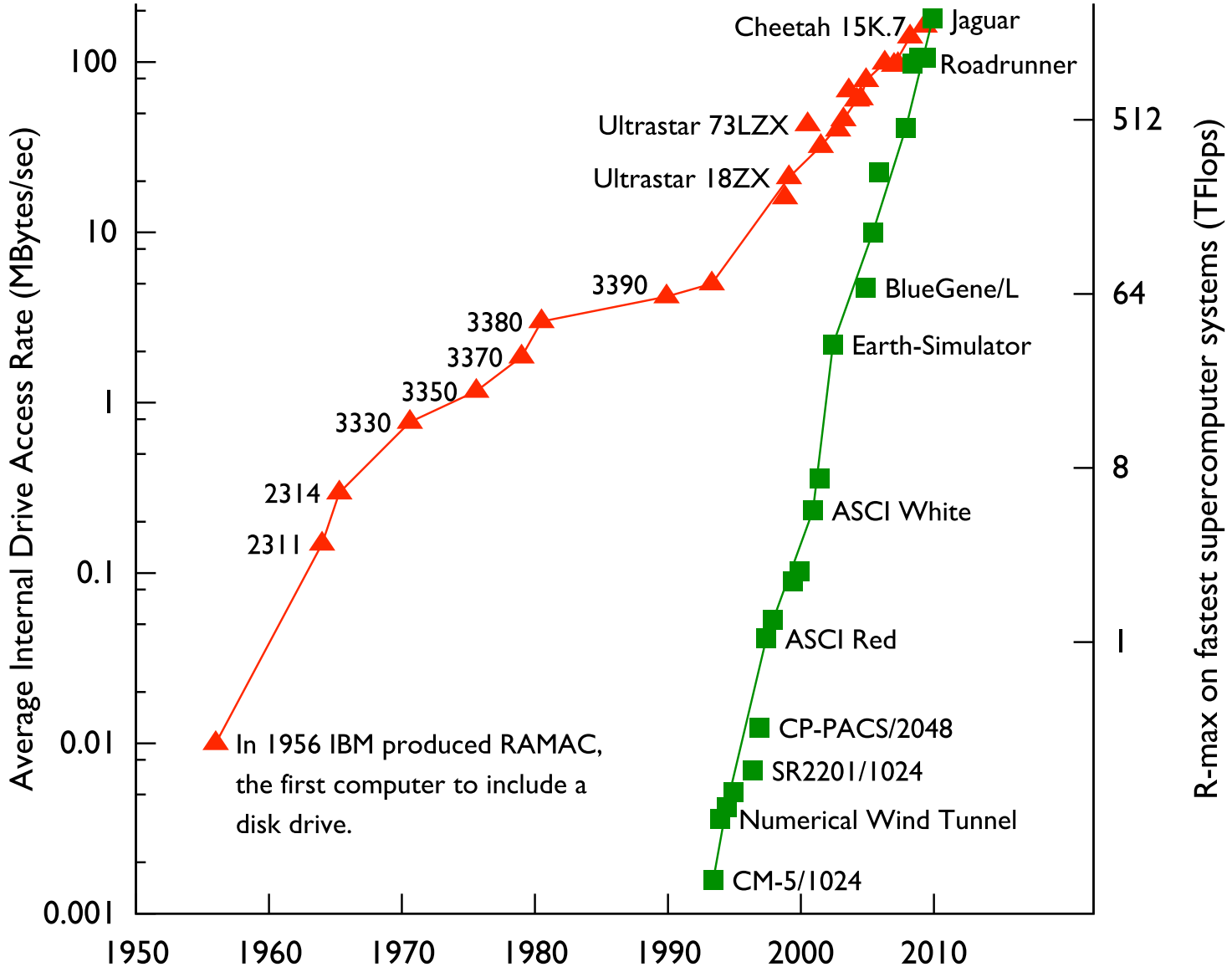
Outline

- The Present
  - Current Leadership storage systems
  - Argonne Blue Gene/P storage study
  - Argonne Blue Gene/P application I/O study
- **The Exascale Era**
  - Roles of I/O at Exascale
  - Performance disparity between I/O and compute
  - Research and development avenues to address gaps
- Conclusions

Roles of Storage in Exascale Systems

- Defensive I/O
  - Storage system used to tolerate failures (i.e., checkpointing)
  - The driving requirement here is a high (perceived) I/O rate, so that the application can quickly return to computation
- Analysis I/O
  - Storage system used to capture results of computation for further study
  - Data reduction often occurs (i.e., relatively “raw” data output from application to save time)
  - Interactive analysis may require timely response of I/O system

Performance Crisis



Succeeding at Exascale

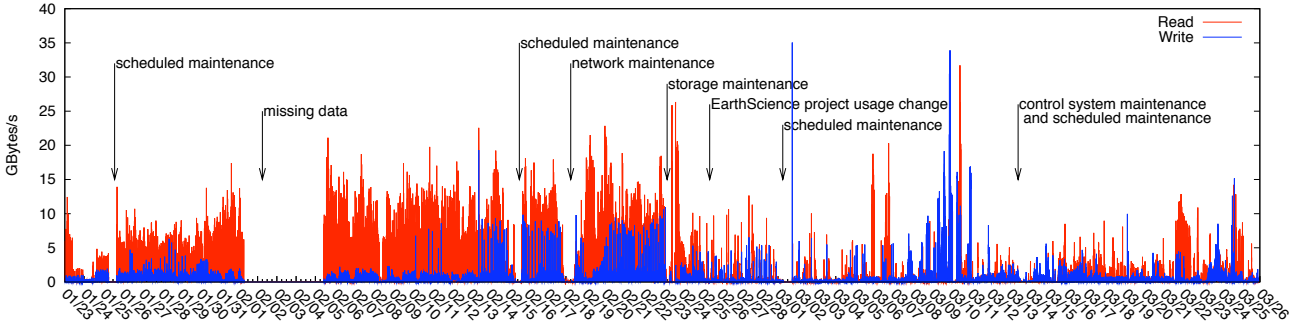
Never waste the opportunities offered by a good crisis.

– Niccolo Machiavelli

- The performance requirements for exascale have people very concerned.
- We must make advances and integrate solutions in (at least) three areas:
  - Enabling aggregation and asynchronous data staging
  - Incorporating techniques for efficient data analysis
  - Applying Data Intensive Supercomputing (DISC) concepts in Exascale HPC

Aggregation and Asynchronous Data Staging

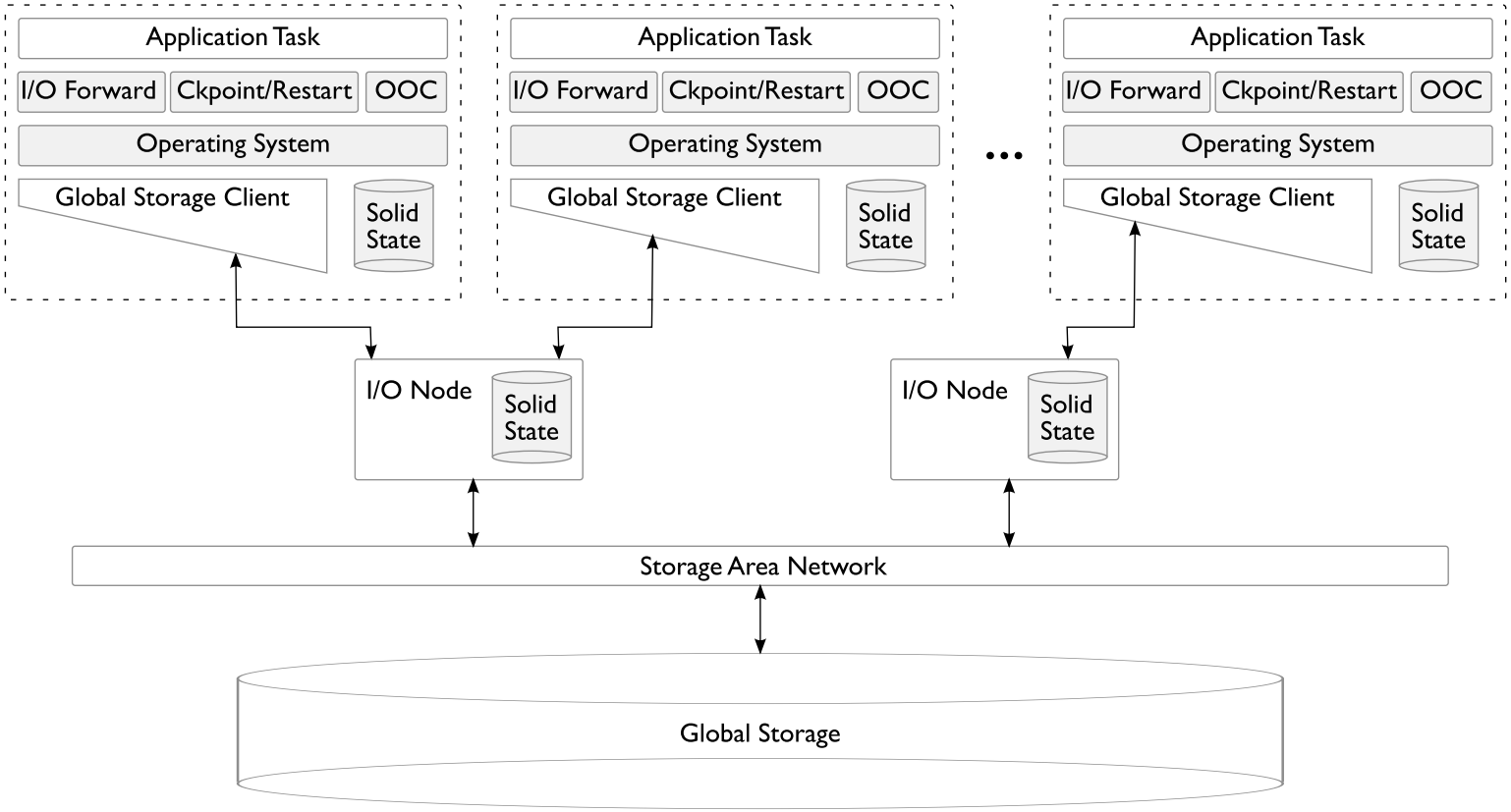
Bursty I/O patterns result in periods of low activity, time we could be moving application data.



- If we could move data asynchronously, we could better use our I/O system and reduce peak I/O demands from applications
  - E.g., by asynchronously staging checkpoints to storage, computation could continue while data moves.
- Questions remain in integrating into exascale systems:
  - Where do we store the data before we have a chance to move it?
  - What software layer is responsible for data movement?
  - How do we drive asynchronous data movement in exascale systems?

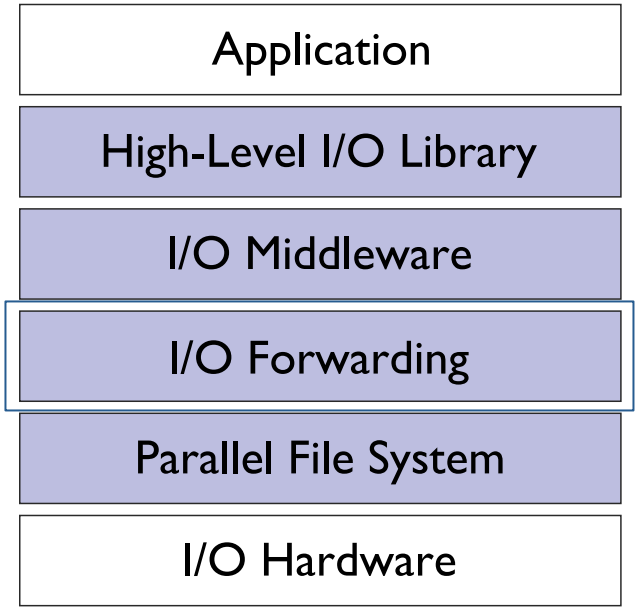
Data Staging in Exascale Systems

- Memory will be extremely limited in exascale systems, so it is unrealistic to consider using it for I/O buffering.
- More likely, solid state storage will be used to provide this staging area.



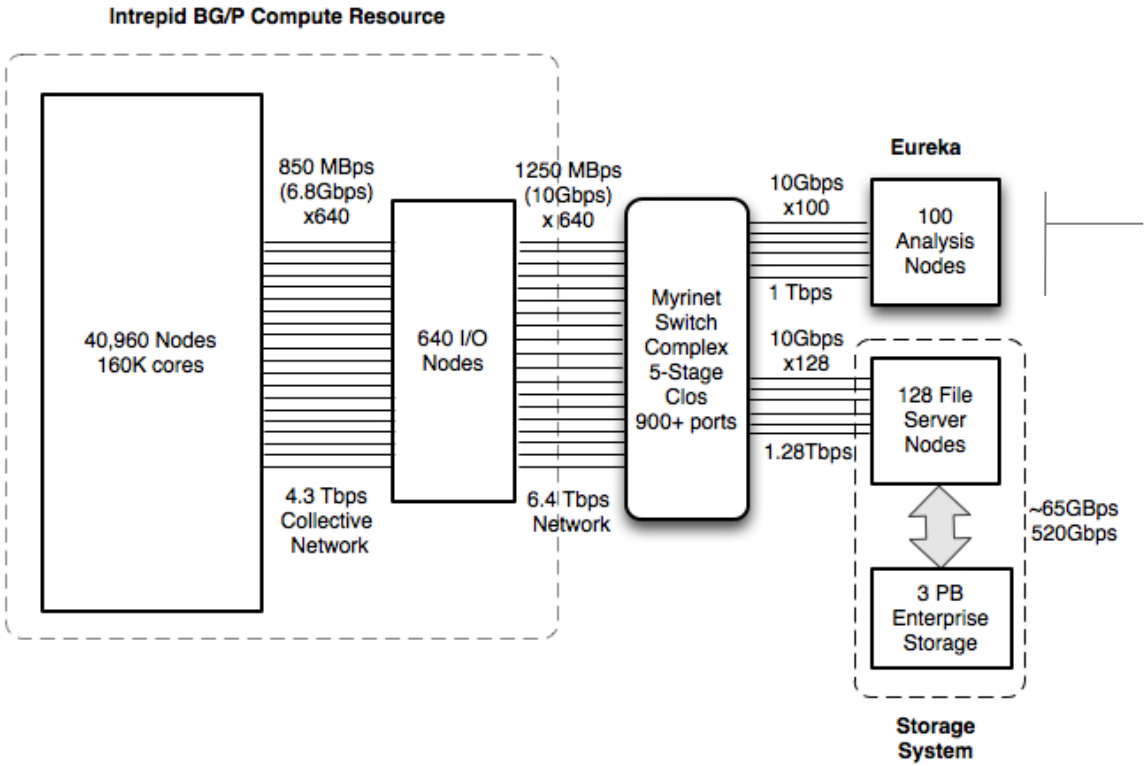
I/O Forwarding and Aggregation

- Provides a bridge between system and storage in machines such as the Blue Gene/P.
- I/O forwarding software is the most encouraging place to perform aggregation and drive asynchronous I/O.
  - All client accesses move through it, unlike MPI-IO and high-level I/O libraries
  - It has hooks into the kernel, providing access to resources
  - It is file system independent
  - It's semantics are underspecified…



N. Ali et al, "Scalable I/O Forwarding Framework for High-Performance Computing Systems," IEEE Cluster, September 2009.

Data Analysis Options



Current system architectures integrate a separate analysis cluster that shares access to storage over a large switch complex. Most data analysis is performed after simulations are complete (post-processed) on these nodes, or processed remotely.

- In situ – process the data (to some degree) in the context of the running application
- Co-processing – process the data around the same time as the simulation is run, but not on the simulation nodes
- Post-processing – store the data and process it later

Image compliments V. Vishwanath (ANL).

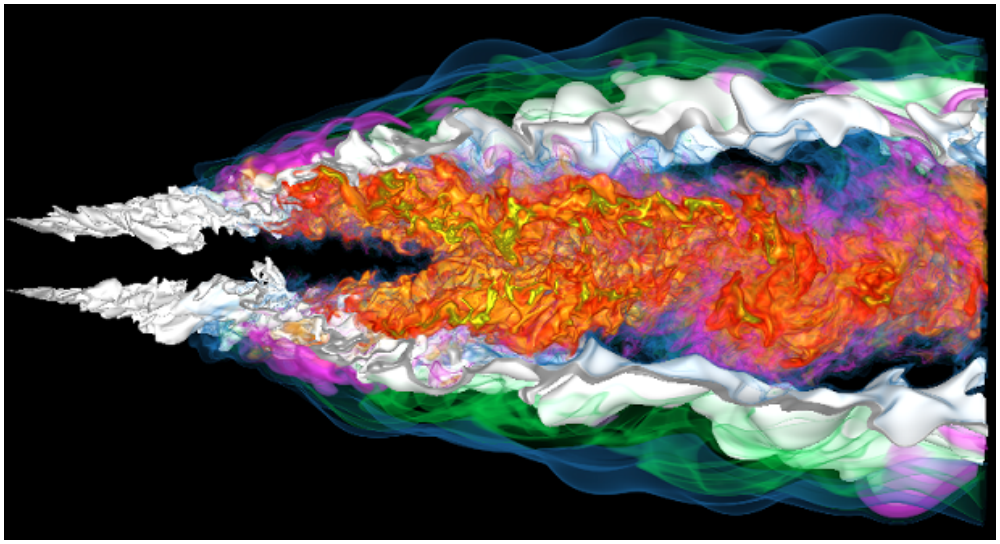
In Situ Analysis and Data Reduction

In situ analysis incorporates analysis routines into the simulation code. This technique allows analysis routines to operate on data while it is still in memory, potentially significantly reducing the I/O demands.

One way to take advantage of in situ techniques is to perform initial analysis for the purposes of data reduction. With help from the application scientist to identify features of interest, we can compress data of less interest to the scientist, reducing I/O demands during simulation and further analysis steps.

The feature of interest in this case is the mixture fraction with an iso value of 0.2 (white surface). Colored regions are a volume rendering of the HO2 variable (data courtesy J. Chen (SNL)).

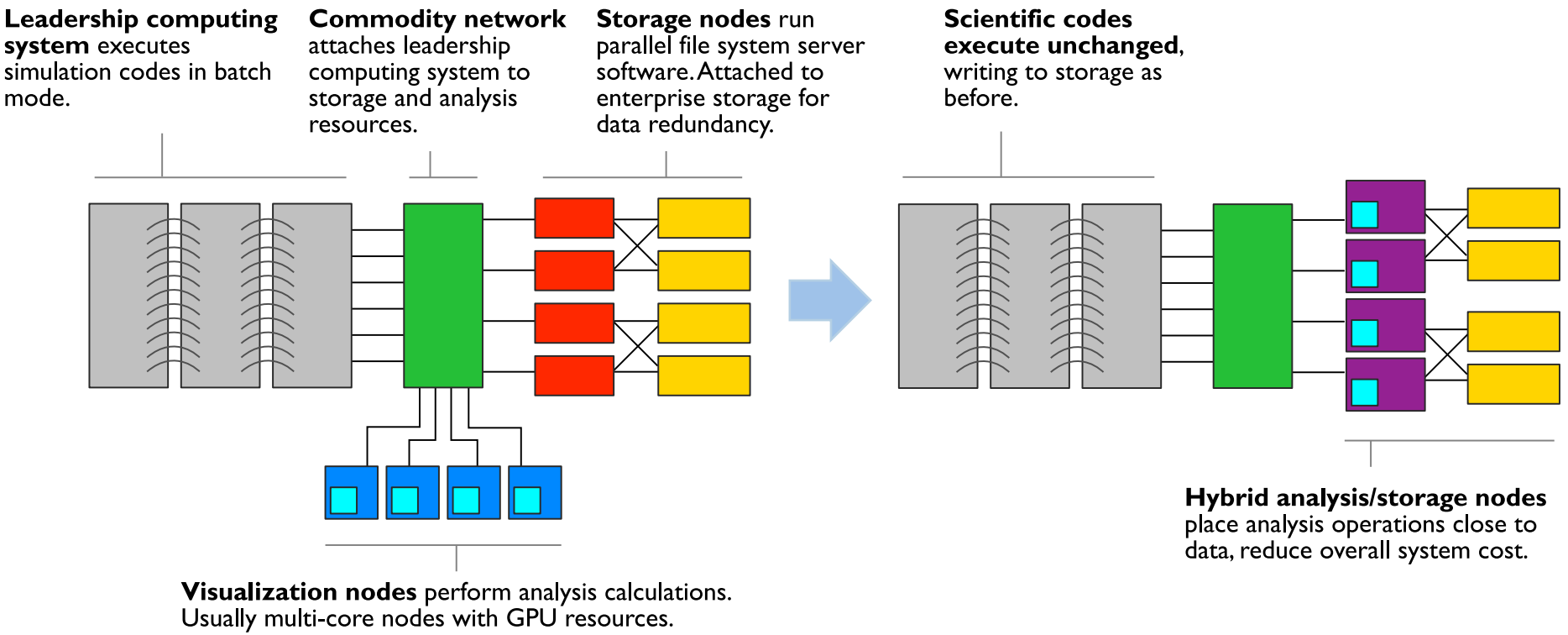
By compressing data more aggressively the further it is from this surface, we can attain a compression ratio of 20-30x while still retaining full fidelity in the vicinity of the surface.



C. Wang, H. Yu, and K.-L. Ma, "Application-driven compression for visualizing large-scale time-varying volume data", IEEE Computer Graphics and Applications, 2009.

Merging Analysis and Storage Resources

One way to reduce costs, and to potentially improve post-processing rates, is to merge analysis resources with storage resources. Need to move to using commodity storage (if possible) at the same time.



Data Intensive Supercomputing (DISC)

- Randal Bryant (CMU) defined DISC in 2007
- Put forth a set of principles for DISC:
  - High-level data-oriented programming model
  - Interactive access -- human in the loop
  - Data as first class citizen
  - Reliability
- First two principles are very relevant to exascale analysis
  - Community already has vtk, ParaView, VisIt, NCL
  - Projects like Scout (McCormick, LANL) also addressing these issues
- Last two principles are very relevant to exascale storage

Randal Bryant, "Data Intensive Supercomputing: The Case for DISC," CMU Technical Report CMU-CS-07-128, May 2007.

Impact of Google on HPC Storage Systems

Internet service companies, and Google in particular, have sparked renewed interest and discussion in large-scale storage system design.

The approach taken by these companies dramatically differs from the traditional HPC architecture:

- Systems built entirely out of commodity hardware
- Software (e.g. Google File System, Chubby, Hadoop) provides resiliency
- Interfaces and semantics (e.g. MapReduce) customized to application set
- Computation mapped onto storage nodes

Benefits:

- Applications easier to develop because APIs match needs
- Application performance improved by coupling with storage

Challenges:

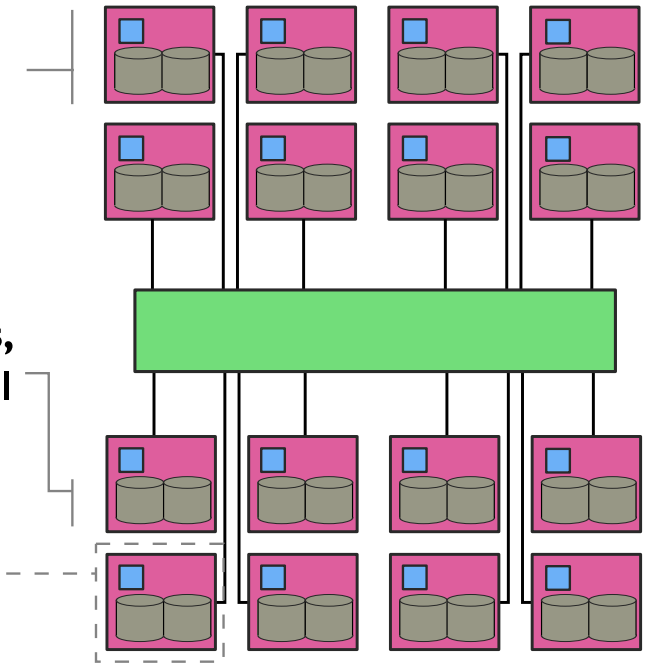
- Applications must be coded to new programming model
- Software must correctly handle a wide variety of failure cases

Google-Style Analysis Clusters

All nodes include storage. Analysis tasks are executed on these nodes and placed to minimize inter-node communication.

Drives are housed in nodes, using commodity SATA or SCSI protocols.

Scalable unit is one node. Bandwidth is 120 MBytes/sec (using Gigabit Ethernet).



Low-cost commodity network (e.g. GigE) used on the assumption that most data will be accessed locally.

Replication between nodes provides fault tolerance.

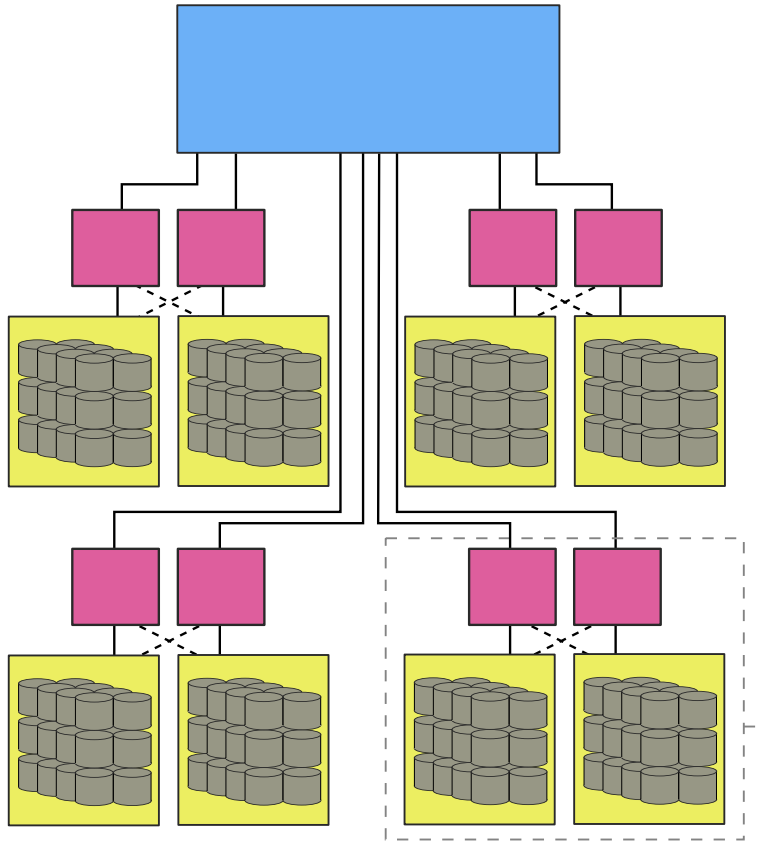
The communication characteristics of a Google-style cluster preclude large-scale science runs, and the number of server nodes needed to achieve HPC bandwidths is many times the number needed in the traditional approach. But **converging on a single system network, better leveraging commodity hardware, and relying on software to manage redundancy are desirable changes.**

An Alternative HPC Storage Model

Storage connected into HPC network, eliminating the need for a separate external network.

Storage nodes manage error correction and run I/O forwarding and file system software. Nodes are aware of underlying structure of storage.

Commodity drive enclosures provide cost-efficient packaging and connectivity. Number of drives chosen to match network bandwidth.



Extension of HPC interconnect allows storage nodes to communicate with one another with low latency and high bandwidth, but **external network connectivity is limited**.

Commodity storage interconnect (e.g. SAS) reduces cost.

Scalable unit is 2 servers and two enclosures. Bandwidth is 4Gbytes/sec.

Incorporating commodity storage enclosures and interconnects significantly reduces cost, while still allowing balance between I/O and network rates. Connecting storage nodes into the HPC network significantly reduces network hops, memory copies, and software layers through which data must pass.

Outline

- The Present
  - Current Leadership storage systems
  - Argonne Blue Gene/P storage study
  - Argonne Blue Gene/P application I/O study
- The Exascale Era
  - Roles of I/O at Exascale
  - Performance disparity between I/O and compute
  - Research and development avenues to address gaps
- **Conclusions**

Conclusions: Co-Design in Exascale Storage

To succeed in developing effective exascale systems, the community must engage in *co-design* of these systems. Co-design incorporates the needs and perspectives of multiple groups into a product (e.g., hardware designers, system software specialists, and application scientists).

- Two needs related to data staging have been pointed out:
  - A low-cost (initial and in power) staging region on nodes
  - Processing capability allocated to drive asynchronous data staging
- Incorporating DISC concepts requires a similar approach.

Parting Thought on Semantics in Storage

- Metadata storage, and data semantics capture in particular, needs more attention.
- Next-generation high-level I/O libraries (replacements for HDF5, PnetCDF, ADIOS) must provide more rich data models for application scientists to use.
- More of the storage system must be made aware of the application data model and how it has been mapped onto storage.
- Semantics of data access need to be relevant to the data model and workflows used by applications, not some arbitrary semantics adopted from some other field.

Acknowledgments

- DOE Office of Advanced Scientific Computing Research (ASCR)
- NSF Office of Cyberinfrastructure
- SciDAC Scientific Data Management Center
- SciDAC Institute for Ultra-Scale Visualization
- SciDAC Center for Scalable Application Development Software

- Pete Beckman, Phil Carns, Jason Cope, Kevin Harms, Kamil Iskra, Dries Kimpe, Sam Lang, Rob Latham, Rusty Lusk, Mike Papka, Tom Peterka, Katherine Riley, Seung Woo Son, Rajeev Thakur, Venkat Vishwanath, Justin Wozniak (ANL)
- Alok Choudhary, Kui Gao, Wei-keng Liao, Arifa Nisar (NWU)
- Kwan-Liu Ma, Hongfeng Yu (UC Davis)
- Chaoli Wang (Michigan Tech)
- Lee Ward (SNL)
- Gary Grider, James Nunez (LANL)
- Steve Poole, Terry Jones (ORNL)
- Yutaka Ishikawa, Kazuki Ohta (University of Tokyo)
- Javier Blas, Florin Isaila (University Carlos III of Madrid)