TEXAS TECH UNIVERSITY™

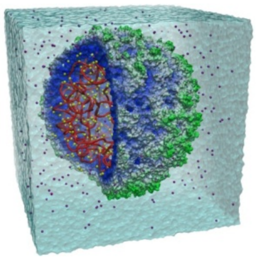
Memory-Conscious Collective I/O for Extreme Scale HPC Systems

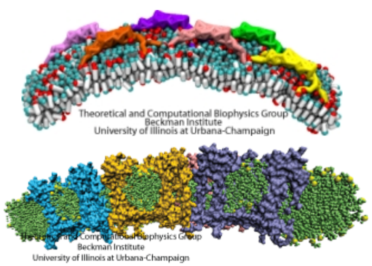
Yin Lu, **Yong Chen**, *Texas Tech University*
Rajeev Thakur, *Argonne National Laboratory*
Yu Zhuang, *Texas Tech University*

Introduction – Data Intensive HPC Simulations/Applications

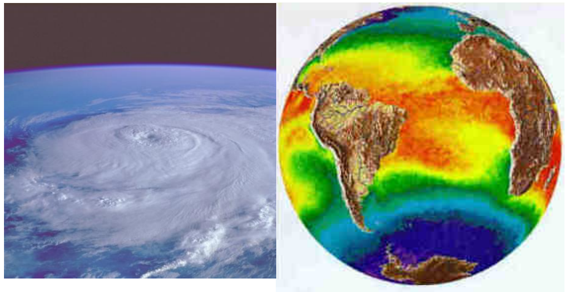
- A wide range of HPC applications, simulations, and visualizations[1]
- Many applications are increasingly data intensive[2]

Molecular Science

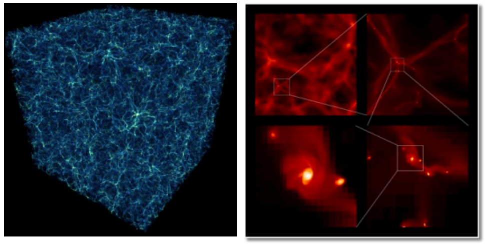




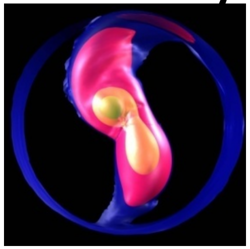
Weather & Climate Forecasting



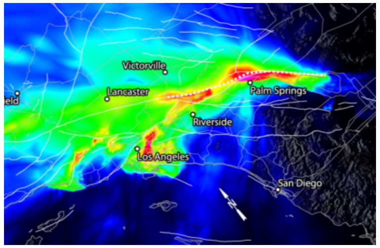
Astrophysics



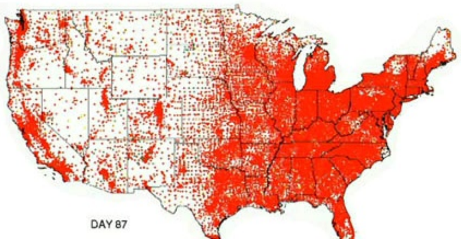
Astronomy



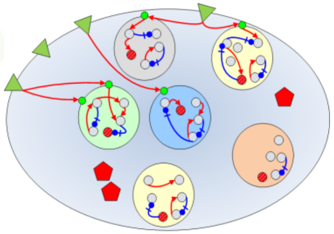
Earth Science



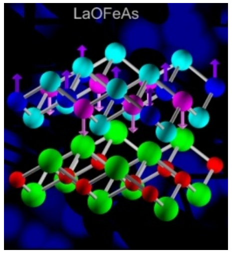
Health



Life Science



Materials



1. Simulation at Extreme Scale, William D. Gropp, Invited presentation at Big Data Science: A Symposium in Honor of Martin Schultz, October 26, 2012, New Haven, Connecticut
2. J.Dongarra, P. H. Beckman, et. al. The International Exascale Software Project roadmap. IJHPCA 25(1): 3-60 (2011)

Introduction – Data Intensive HPC Simulations/Applications

- Many simulations/applications process O(1TB-100TB) in a single run
- Application teams are projected to manipulate O(1PB-10PB) on exascale systems

Data requirements for representative INCITE applications at ALCF

PI	Project	On-Line Data	Off-Line Data
Lamb, Don	FLASH: Buoyancy-Driven Turbulent Nuclear Burning	75TB	300TB
Fischer, Paul	Reactor Core Hydrodynamics	2TB	5TB
Dean, David	Computational Nuclear Structure	4TB	40TB
Baker, David	Computational Protein Structure	1TB	2TB
Worley, Patrick H.	Performance Evaluation and Analysis	1TB	1TB
Wolverton, Christopher	Kinetics and Thermodynamics of Metal and	5TB	100TB
Washington, Warren	Climate Science	10TB	345TB
Tsigelny, Igor	Parkinson's Disease	2.5TB	50TB
Tang, William	Plasma Microturbulence	2TB	10TB
Sugar, Robert	Lattice QCD	1TB	44TB
Siegel, Andrew	Thermal Striping in Sodium Cooled Reactors	4TB	8TB
Roux, Benoit	Gating Mechanisms of Membrane Proteins	10TB	10TB

Source: R. Ross et. al., Argonne National Laboratory

Motivation – Decreased Memory & BW per core at Exascale

- Neither available memory capacity nor memory bandwidth scales by the same factor as the total concurrency
- The disparity of growth between memory and concurrency indicates the average memory and bandwidth per core even drop in exascale system

U.S. DEPARTMENT OF ENERGY

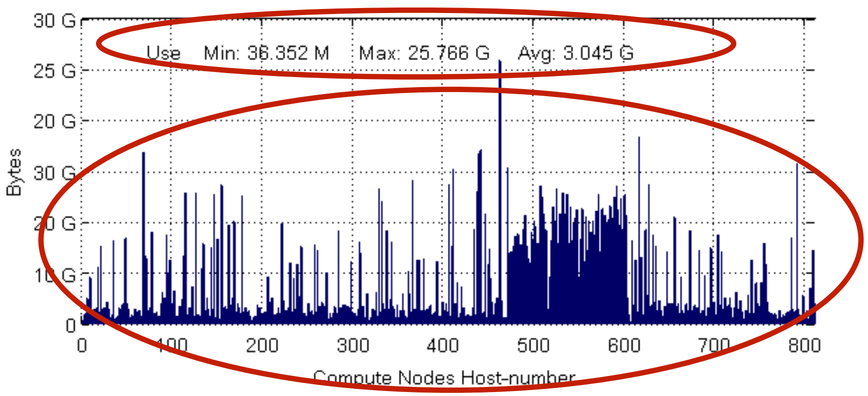
Expected Exascale Architecture Parameters and Comparison with Current Hardware[3]

System Parameter	2011	2018	Factor Change
System Peak	2 Pf/s	1 Ef/s	500
Power	6 MW	≤20 MW	3
System Memory	0.3 PB	10 PB	33
Total Concurrency	225 K	1B	4444
Node Performance	0.125 Tf/s	1 Tf/s	80
Node Memory BW	25 GB/s	400 GB/s	16
Node Concurrency	12 CPUs	1000 CPUs	83
Interconnect BW	1.5 GB/s	100 GB/s	66
System Size (nodes)	18700	1000000	50
I/O capacity	15 PB	300 PB – 1000 PB	20 - 67
I/O Bandwidth	0.2 TB/s	20 – 60 TB/s	10 -30

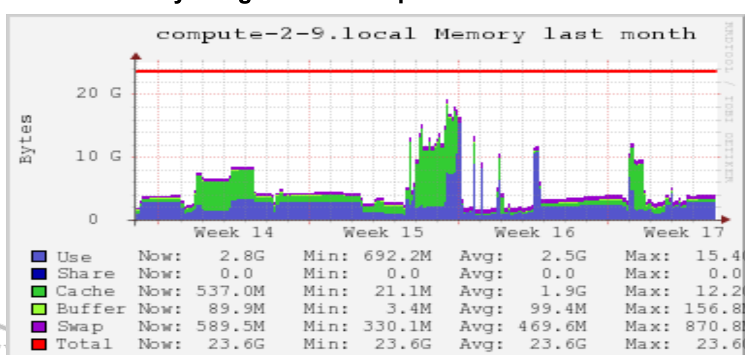
3. S. Ahern, A. Shoshani, K.-L. Ma, et al. Scientic discovery at the exascale. Report from the DOE ASCR 2011 Workshop on Exascale Data Management, Analysis, and Visualization, February 2011

Motivation – Increased Available Memory Variance

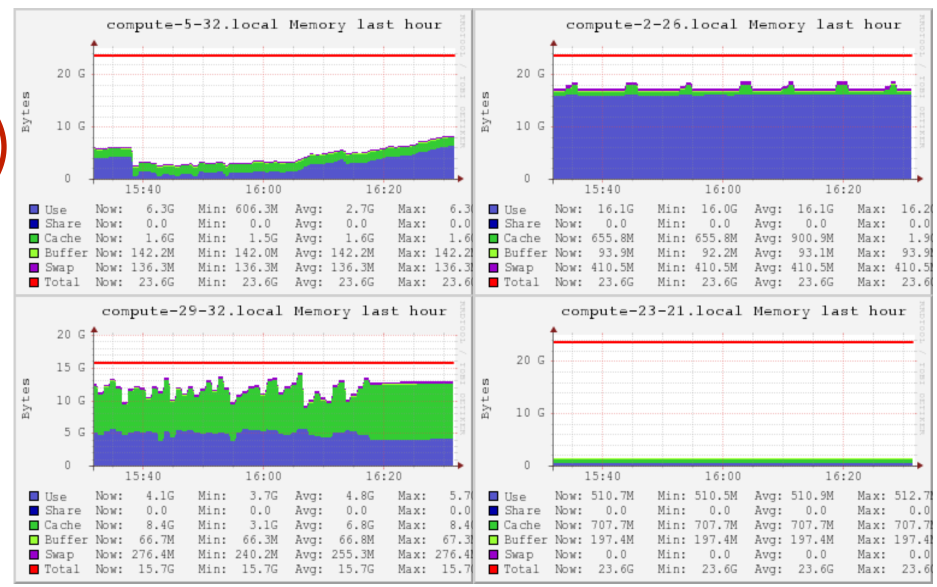
- Available memory exhibits imbalance among compute nodes
- Available memory per node can vary significantly at an extreme scale
- These projected constraints present challenges for I/O solution at exascale including collective I/O



Memory Usage of 815 compute nodes at one time



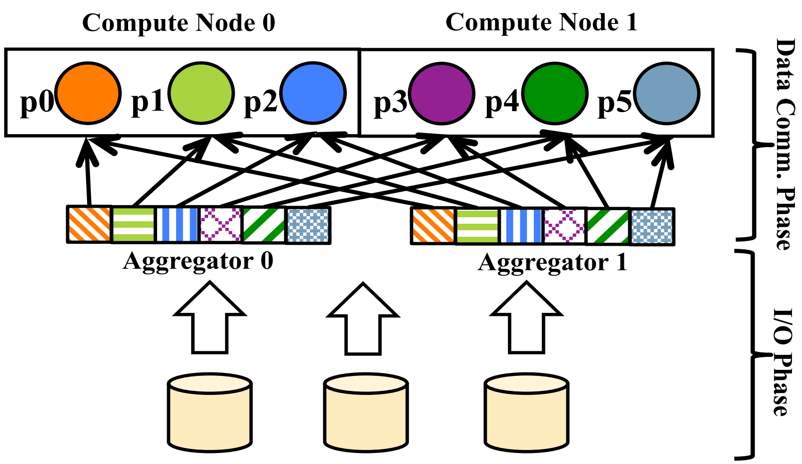
Memory Usage of a single compute node in one month

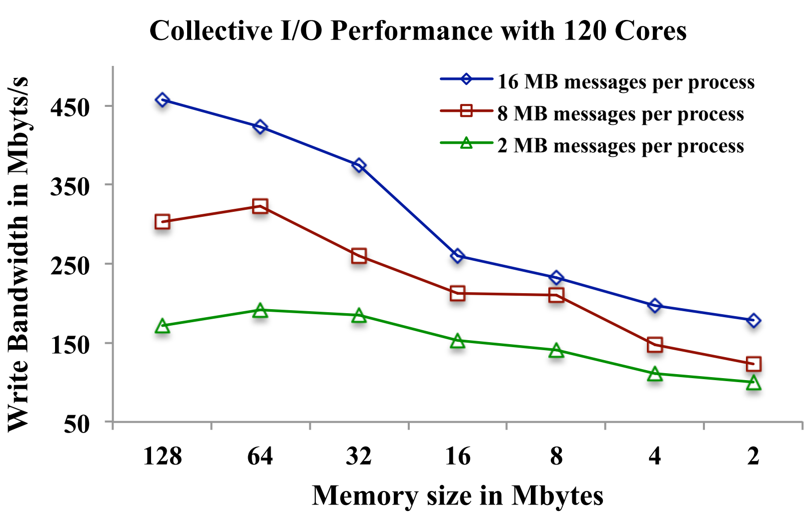


Memory Usage of four compute nodes in one hour

Motivation – Collective I/O Performs with Limited Memory

- *Collective I/O* optimizes I/O accesses by merging small & noncontiguous I/O requests into large & contiguous ones, removing overlaps, reducing calls
- Remains critical for extreme scale HPC systems
- Performance can be significantly affected under memory pressure





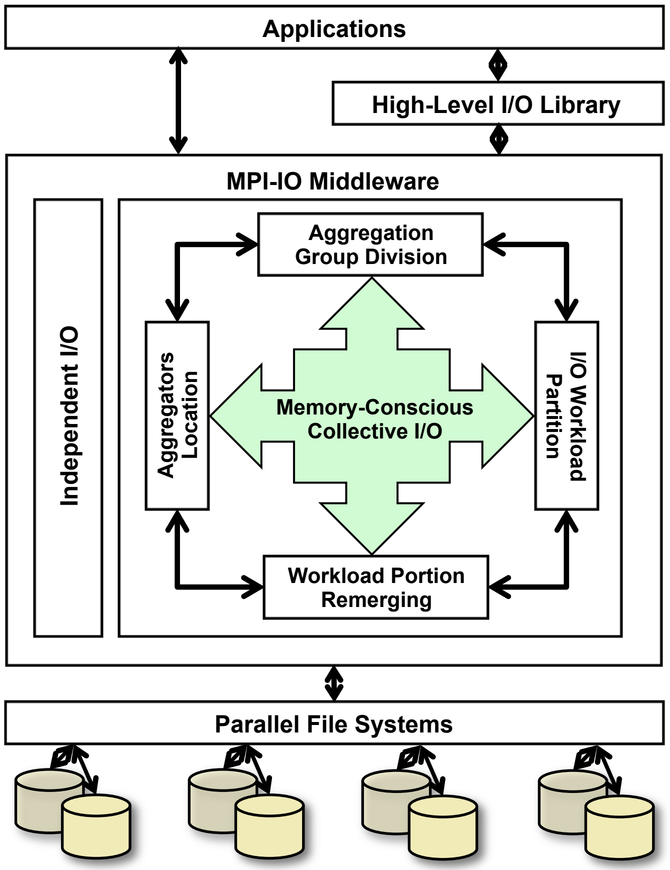
Performance of Collective I/O for Various Memory Sizes

Memory-Conscious Collective I/O

- Objective: to design and develop collective I/O with awareness of memory capacity, variance, off-chip bandwidth
- Contributions
  - Identified performance & scalability constraints imposed by memory pressure issue
  - Proposed a memory-conscious strategy to conduct collective I/O with memory-aware data partition and aggregation
  - Prototyped and evaluated the proposed strategy with benchmarks
  - Memory-conscious strategy can be important given the significance of collective I/O and substantial memory pressure at extreme scale
  - Towards addressing challenges of an exascale I/O system

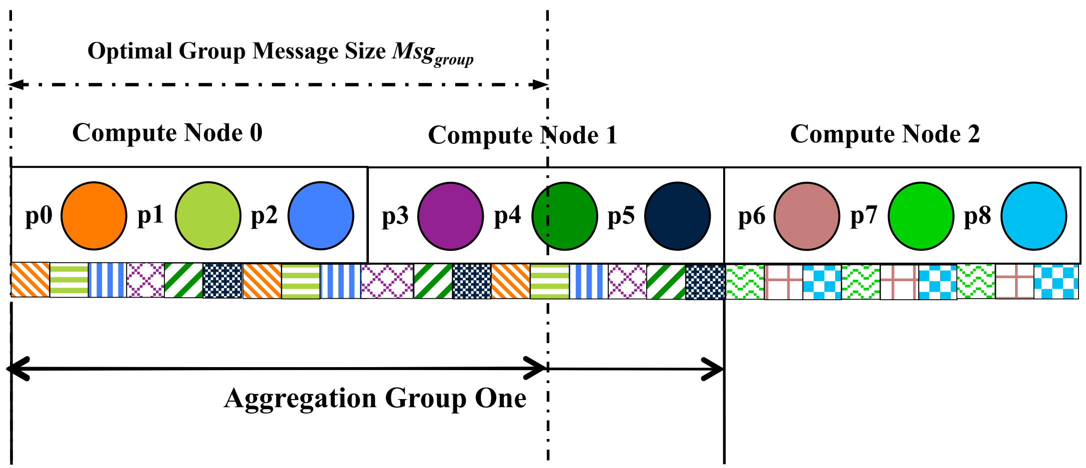
Memory-Conscious Collective I/O (cont.)

- Contains four major components
- *Aggregation Group Division* divides the I/O requests into separated groups
- *I/O Workload Partition* partitions the aggregate access file region into contiguous file domains
- *Workload Portion Remerging* rearranges the file domains considering the memory usage of physical nodes
- *Aggregators Location* determines the placement of aggregators for each file domain
- Applications, library, parallel file systems



Aggregation Group

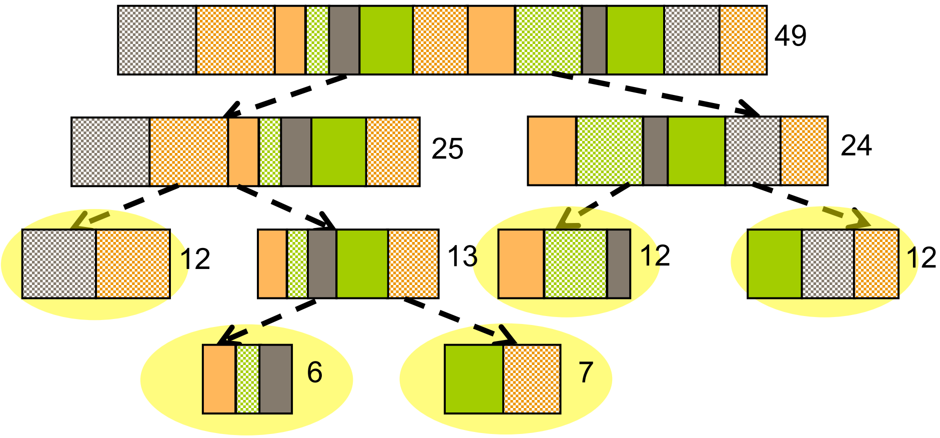
- To avoid global aggregation and reduce memory requirements
- The global data shuffling traffic increases the memory pressure on aggregators and leads to off-chip memory bandwidth contention
- Divides the I/O workloads into non-overlapping chunks guided by the optimal group message size Msg_{group}
- Aggregation groups perform their own aggregation in parallel
- Limit one node in one group to reduce communications



I/O Workload Partition within Aggregation Group

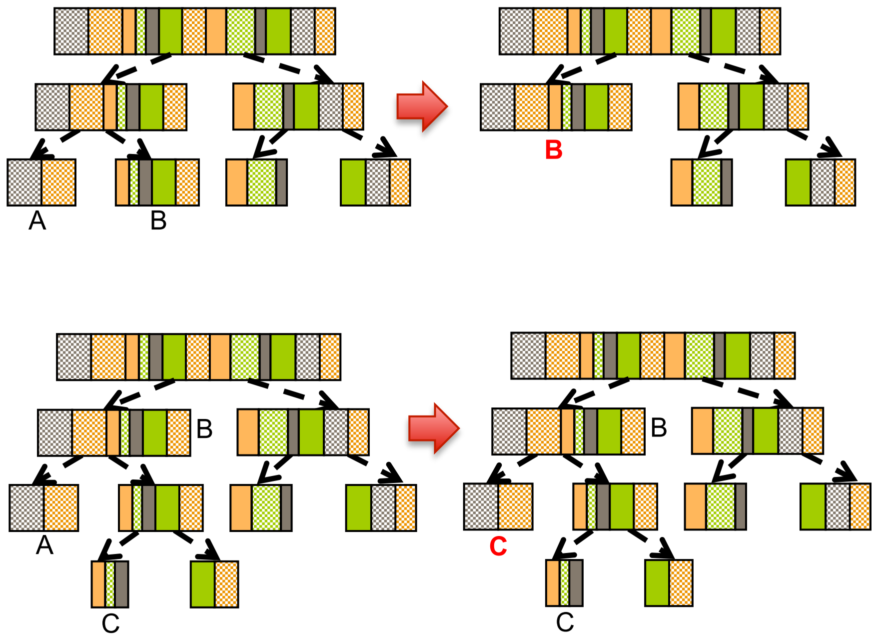
- Analyzes all I/O accesses within each aggregation group
- Workload dynamically partitioned into distinct domains

```
Dynamical Workload Partition Algorithm
Bisect
{
    Compute root weight Root_wgh ;
    If Root_wgh > Msg_ind
            Bisect_tree(root);
}
Bisect_tree(vertex)
{
    Create two children for the vertex;
    Split the region belonging to this vertex in half;
    The compute nodes with associated messages in this
region are partitioned accordingly into two sets;
    Assign each set to one child;
    For each child
    {
            Compute child weight Child_wgh;
            If Child_wgh > Msg_ind
                    Bisect_tree(child);
    }
}
```



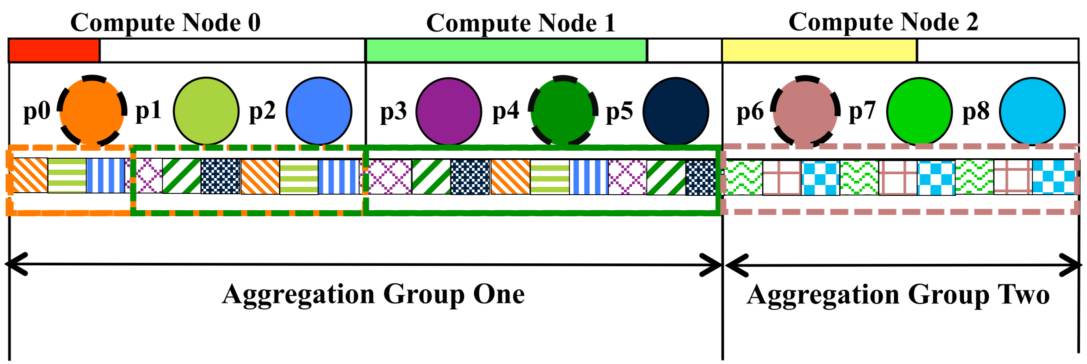
Workload Portions Remerging

- Reorganizes the file domains considering the memory consumption for the aggregation
- File domain merged with the domain nearby (still a contiguous file domain)
- To aggregate I/O requests based on available memory & saturate B/W



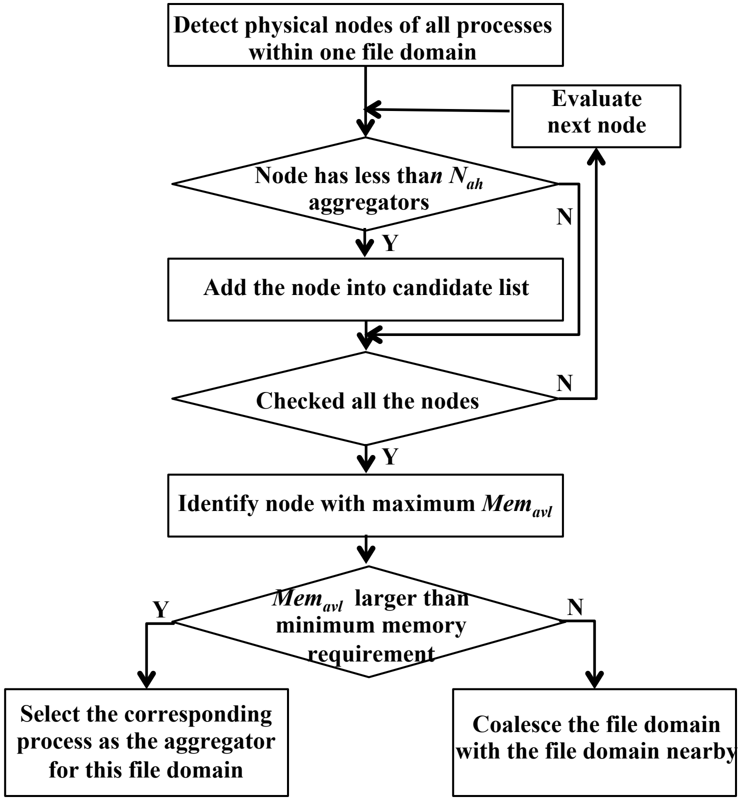
An Example and Comparison

- Aggregation and file domain partition with memory-conscious strategy
- Compared against conventional evenly partition
- Avoid iterations of carrying out collective I/O



Aggregators Location

- Limits the number of aggregators in a node
- Identifies the node with required available memory and minimizes communications and B/W requirement

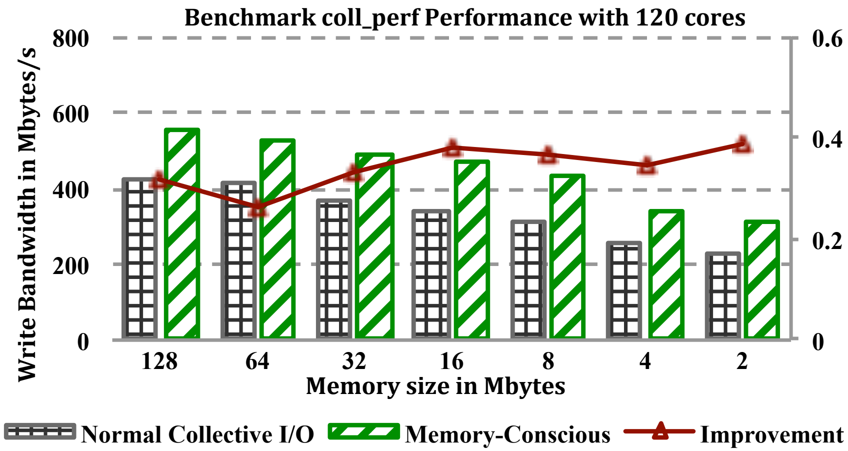


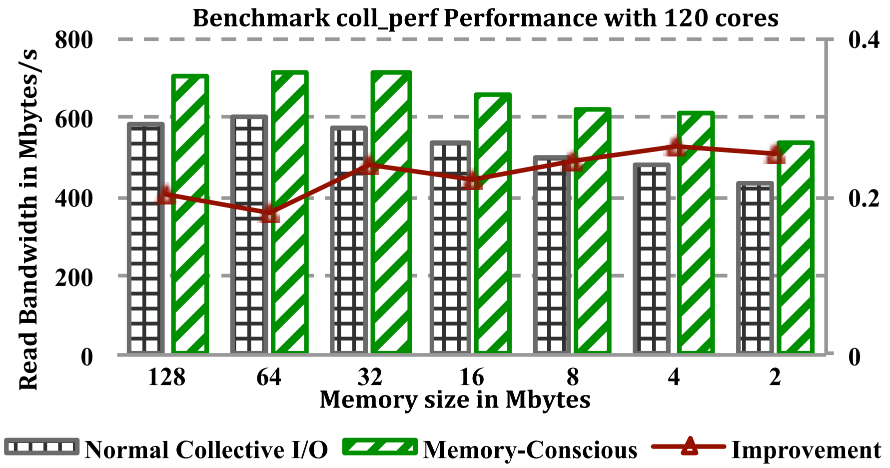
Evaluation and Performance Analysis

- Experimental Environment
  - 640-node Linux-based cluster test bed with 600TB Lustre file system
  - Each node contains two Intel Xeon 2.8 GHz 6-core processors with 24 GB main memory
  - Nodes connected with DDR InfiniBand interconnection
  - Prototyped with MPICH2-1.0.5p3 library
- Three well-known MPI-IO benchmarks selected for evaluation & comparison against normal collective I/O
  - coll_perf from ROMIO software package
  - IOR developed at Lawrence Livermore National Laboratory
  - MPI-IO Test developed at Los Alamos National Laboratory

Evaluation and Performance Analysis

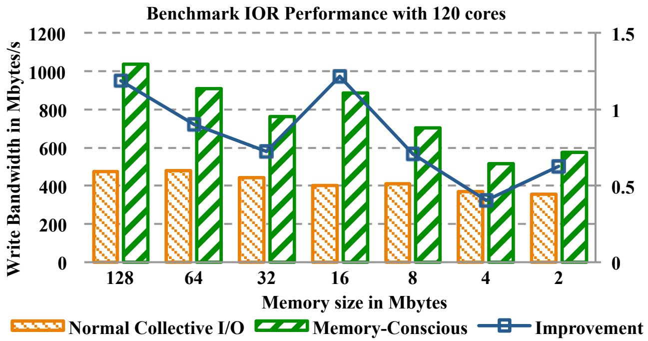
- Experimental Results of coll_perf Benchmark
  - 120 MPI processes used to write and read a 32 GB file on Lustre file system
  - Evicted cached data with memory flushing after write phase
  - Average performance for write and read tests were 34.2% and 22.9% respectively

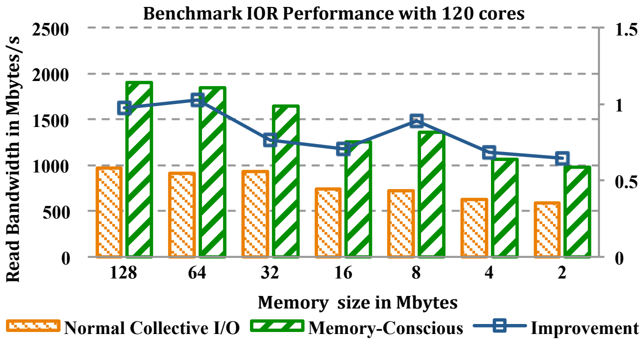


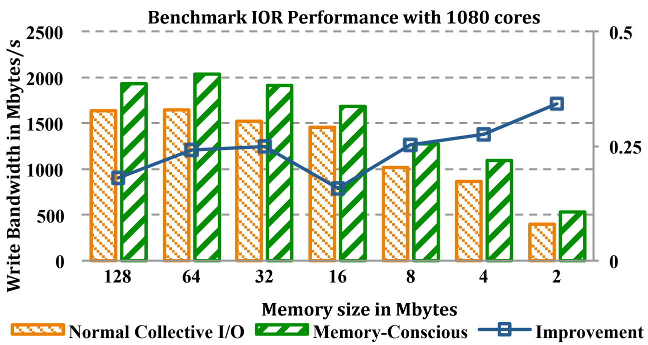


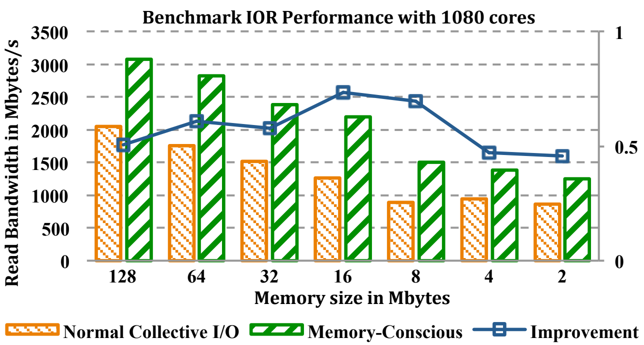
Evaluation and Performance Analysis

- Experimental Results of IOR Benchmark
  - Tests carried out with 120 and 1080 processes
  - Maximum write and read improvement up to 121.7% and 89.1% respectively
  - Write tests performance improvements were more sensitive to the new strategy
  - Average performance for write and read tests were 24.3% and 57.8% respectively



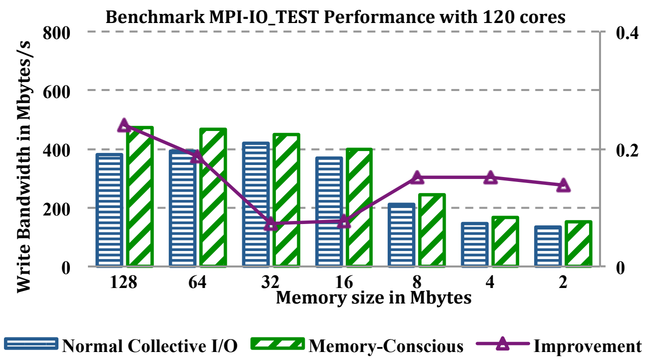


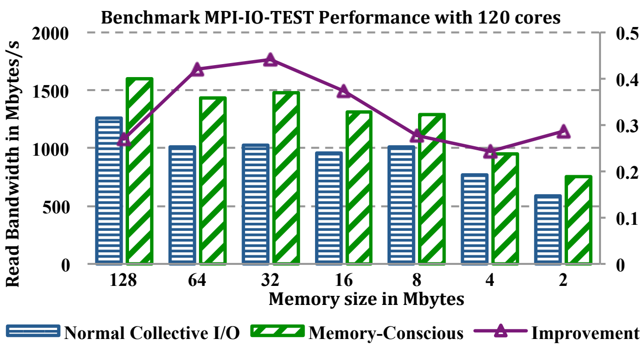


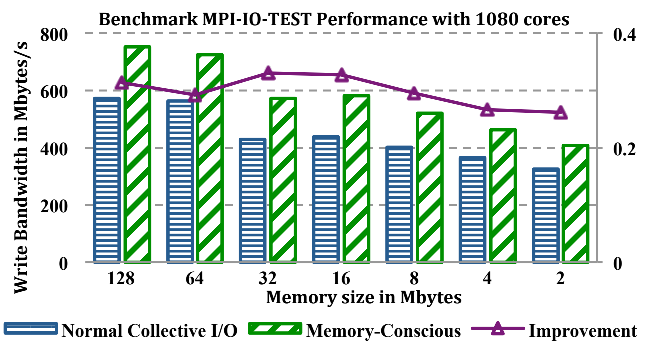


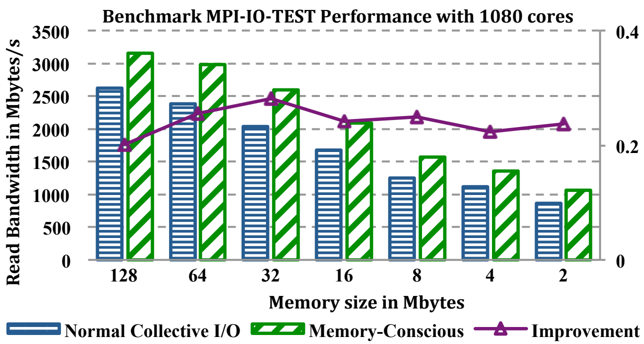
Evaluation and Performance Analysis

- Experimental Results of mpi-io-test Benchmark
  - Performance increased at a relatively moderate rate compared with IOR
  - Average performance improvements for read and write tests were 32.9% and 14.6% at 120 cores
  - Average performance improvements for read and write tests were 29.8% and 24.1% at 1080 cores









Conclusion

- Exascale HPC systems near the horizon
  - Decreased memory capacity per core, increased memory variance, and decreased bandwidth per core are critical challenges for collective I/O
- Studied the constraints at projected exascale systems
- Proposed a new memory-conscious collective I/O strategy
  - Restricts aggregation data traffic within groups
  - Determines I/O aggregation dynamically
  - With memory-aware data partition and aggregation
- Experiments performed on MPICH2+Lustre
- Evaluation results confirmed the proposed strategy outperformed existing collective I/O given memory constraints

Future Work

- An I/O system aware of memory constraints can be critical on current petascale and projected exascale systems
- The memory-conscious collective I/O strategy
  - Retains benefits of collective I/O
  - Reduces memory pressure
  - Alleviates off-chip bandwidth contention
- Plan to further investigate and reduce communication costs
- Plan to investigate the leverage of SCM, burst buffer, caching

Any Questions?

Thank You.

For more information please visit

http://discl.cs.ttu.edu/