Persistent Memory and I/O Middleware

Quincey Koziol
Lawrence Berkeley National Laboratory

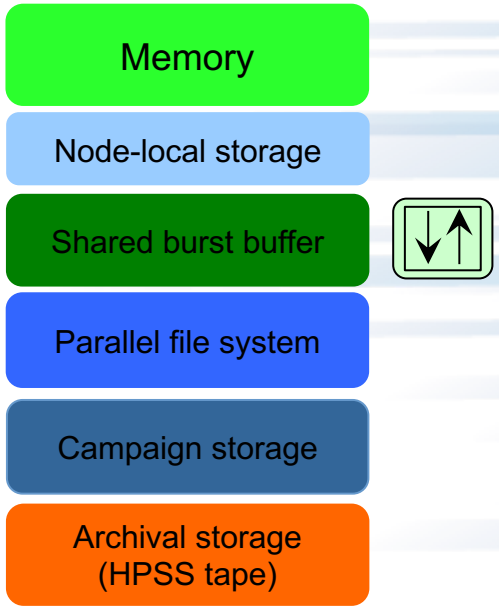
Houston, TX
Feb 6th, 2020

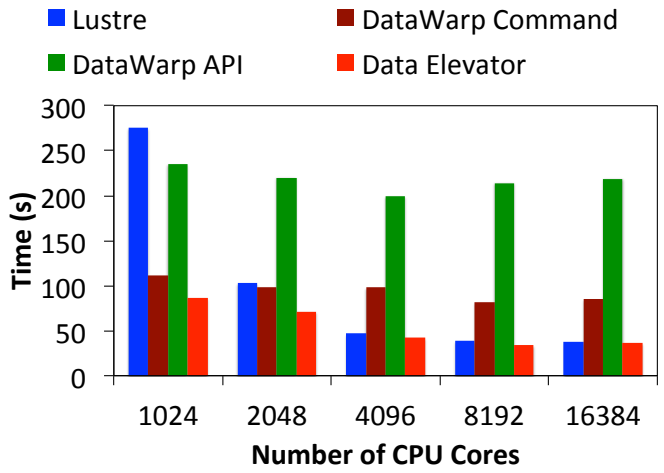
Acknowledgements

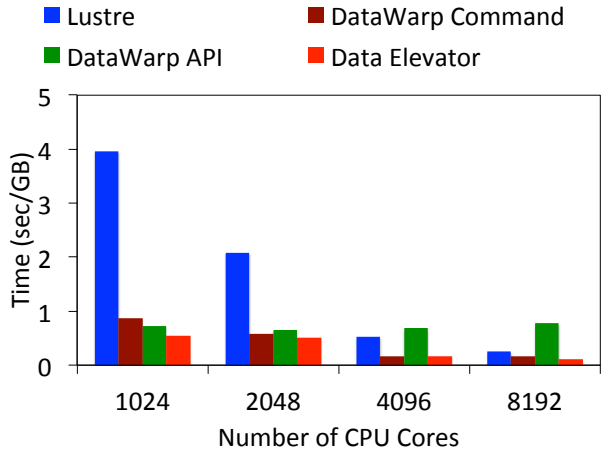
- **LBNL**: Suren Byna, Houjun Tang, Bin Dong, Tonglin Li
- **The HDF Group**: Elena Pourmal, Scot Breitenfeld, Jerome Soumagne, Richard Warren, Kimmy Mu, Dana Robinson
- **ANL**: Venkat Vishwanath, Huihuo Zheng, Paul Coffman
- *And others!*

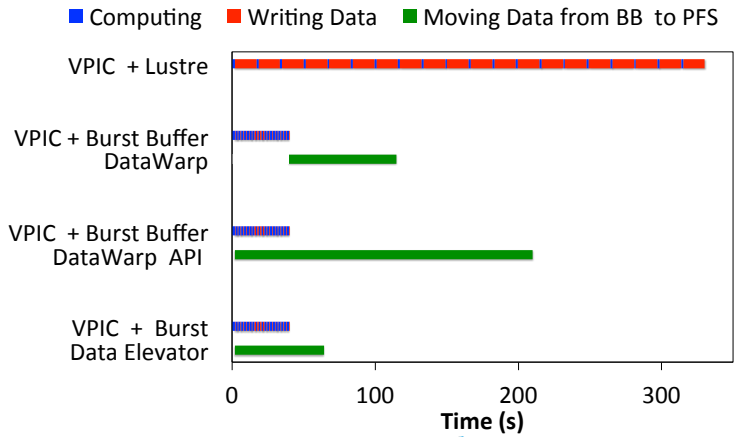
Features: Data Elevator for using shared burst buffers - Write

- Data Elevator write caching using burst buffers
  - Transparent data movement in storage hierarchy
  - In situ data analysis capability using burst buffers
- Tested with a PIC code and Chombo-IO benchmark
- Applications evaluating Data Elevator
  - E3SM-MMF and Sandia ATDM project is evaluating performance
  - Other candidates: EQSim, AMReX
- Installed on NERSC's Cori system (*module load data-elevator*)



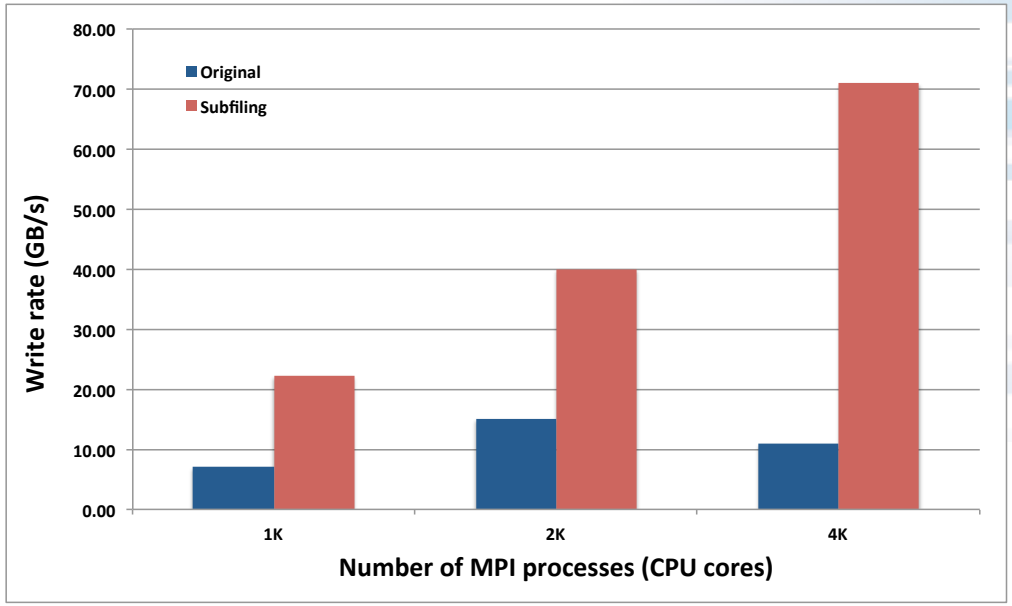






Features: Sub-filing

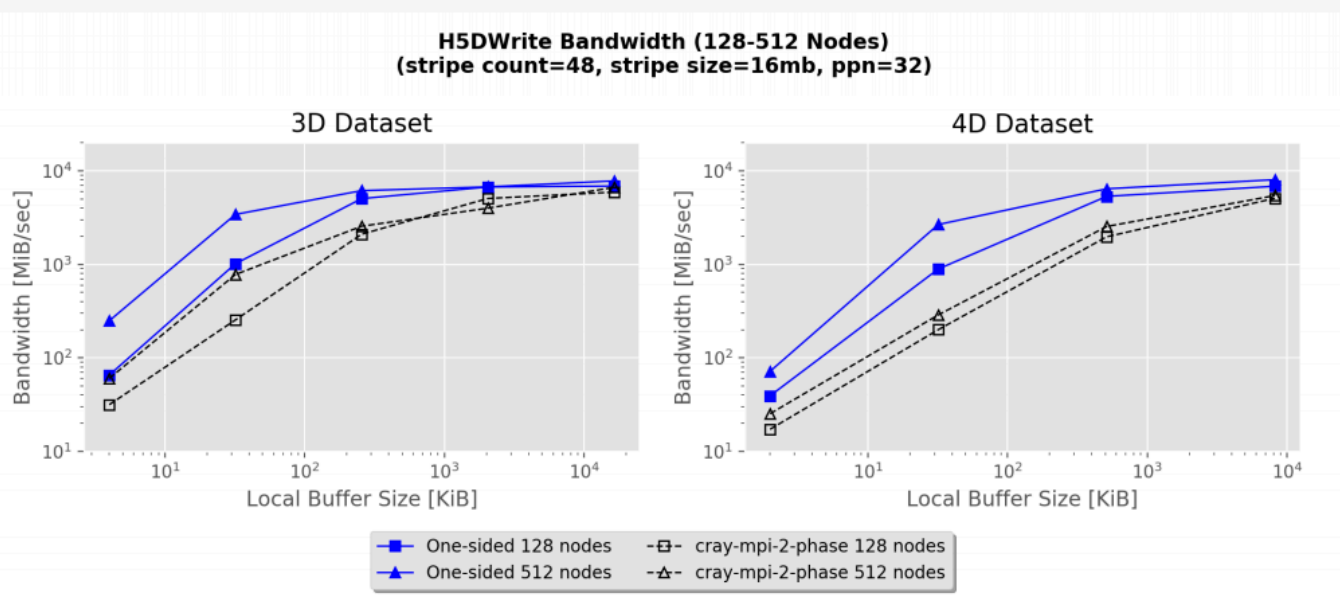
- Writing to single shared file is slow due to:
  - Locking contention
- A solution: Sub-filing
  - Multiple small files
  - A metadata file stitching the small files together
- Benefits
  - Better use of parallel I/O subsystem
  - Reduced locking and contention issues improve performance
- Designing production quality implementation of sub-filing in HDF5 using Virtual File Driver (VFD)
  - Will use node-local storage for caching



Performance on Cori with prototype implementation to show the potential of sub-filing

Features: System topology-aware VFD

- Taking advantage of the topology of compute and I/O nodes and network among them improves overall I/O performance
- Developing topology-aware data-movement algorithms and collective I/O optimizations within a new HDF5 virtual file driver (VFD)



Performance comparison of the new HDF5 VFD, using one-sided aggregation, with the default binding to Cray MPICH MPI-IO. Data was collected on Theta using an I/O benchmarking tool (the HDF5 Exerciser),

Prototype implementation: **CCIO** branch https://bitbucket.hdfgroup.org/projects/HDFFV/repos/hdf5/

Proactive Data Containers (PDC): A Portable Object Data Management System for HPC systems

Scientific Achievement

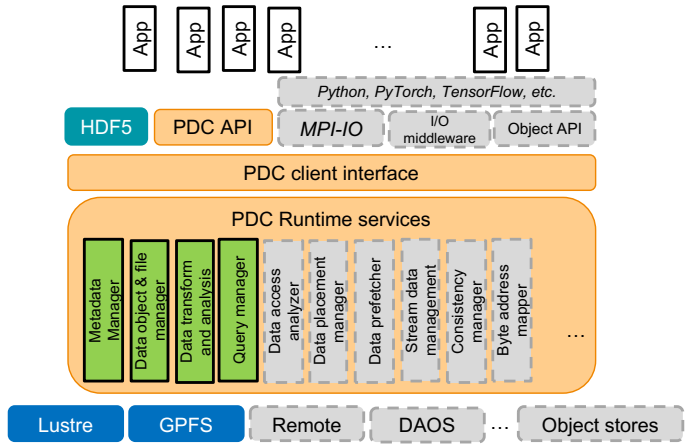
PDC achieves efficient storage and access of data with simple object abstractions, transparently taking advantage of deep and heterogeneous HPC storage hierarchy.

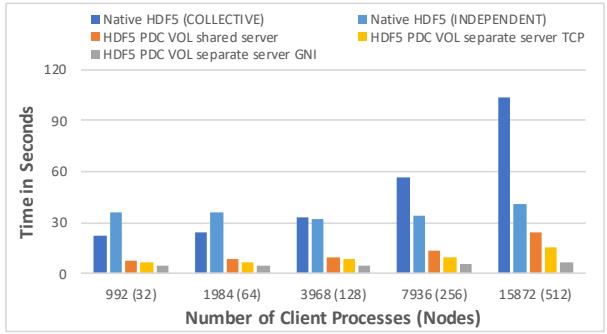
Significance and Impact

Applications store data as PDC objects, which the PDC runtime system transparently and efficiently manages in the storage hierarchy. PDC is portable over underlying HPC file systems, so users don't need special installations.

Research Details

- Existing data management and I/O solutions are based on POSIX semantics and face performance challenges
- We developed a novel data management system with simple data object interfaces, efficient and transparent data movement in storage hierarchy, proactive analysis in the data path, and scalable metadata management
- PDC object management outperforms highly-tuned POSIX I/O based on HDF5 by up to **8X** for writing and by up to **22X** for reading. Searching metadata is **40X** faster than Lustre file system
- Developed a connector for HDF5 applications to use PDC





PDC Overview and Performance: Top figure shows an overview of PDC interfaces through HDF5 and PDC's object interface and various PDC services. Gray boxes show future work. The bottom figure shows the performance of writing particle data (from 248GB to ~4TB) with HDF5 and different configurations of PDC. PDC outperforms highly-tuned POSIX I/O with HDF5 by **6.5X** on average.

Work was performed at LBNL, the HDF Group, and ANL

K. Mu, J. Soumagne, S. Byna, "Interfacing HDF5 with A Scalable Object-centric Storage System on Hierarchical Storage", Concurrency and Computation: Practice and Experience (CCPE) journal