

An Application Driven I/O Optimization Approach for PetaScale Systems & Scientific Discoveries

Wei-keng Liao, Alok Choudhary, Kui Gao, Arifa Nisar, Chen Jin
Northwestern University

Supported by
NSF: HECURA, SDCI HPC
DOE: SciDAC SDM center

Collaborators:
Rob Latham, Rob Ross (ANL), Chris Daley (ASC Flash Center)
Karen Schuchardt, Bruce Palmer, Annette Koontz (PNNL)

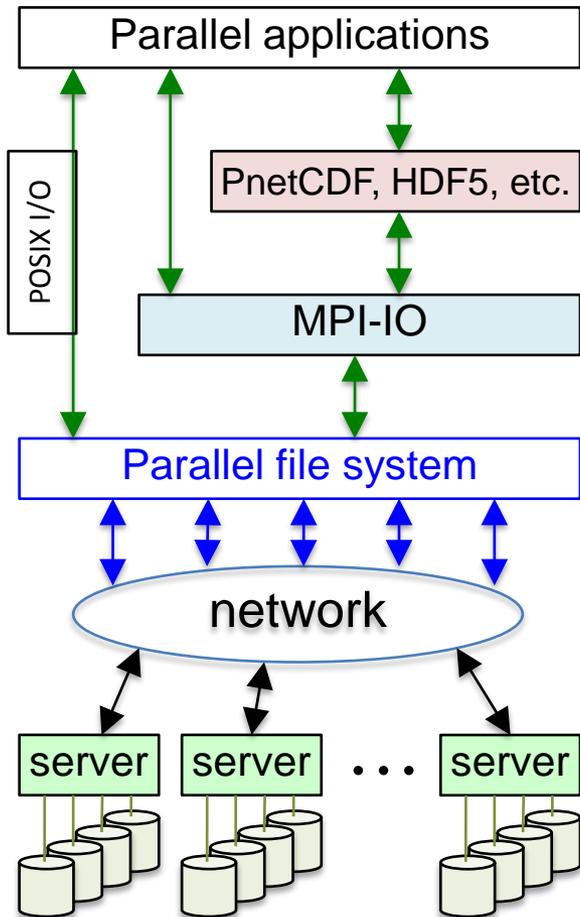
Outline

- Study application I/O patterns
 - Know the intent, not the offsets and bytes
- Study parallel I/O strategies
 - High-level libraries (eg. PnetCDF, HDF5)
 - Middleware (MPI-IO)
 - Parallel file system (striping)
- Proposed work
 - Optimizations in PnetCDF and MPI-IO
 - An I/O benchmark

Understanding application I/O patterns

- Large-scale scientific computer simulations
 - Use high-level I/O libraries, such as PnetCDF, HDF5
 - Portable and descriptive
 - I/O operations are variable-based
 - Contiguous/non-contiguous accesses
- Application I/O kernels under study
 - S3D (block partitioned data)
 - FLASH3 (AMR, block based I/O)
 - Chombo (AMR, irregular-sized, unbalanced I/O)

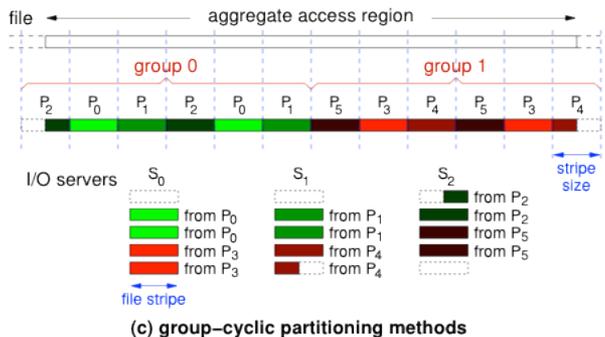
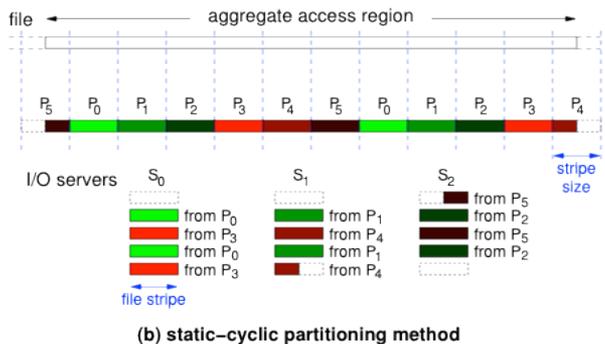
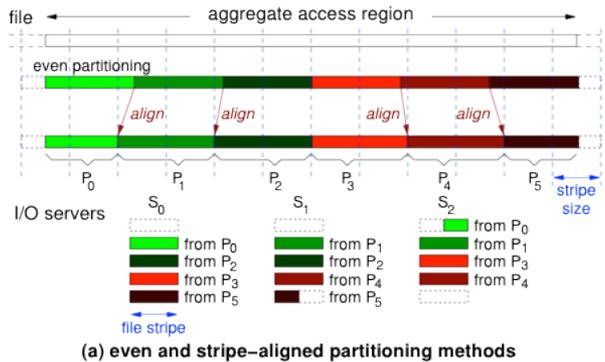
High-level I/O libraries



- Parallel netCDF, NetCDF4, HDF5
 - Define variables with descriptive information
 - A variable ID is used as an argument in any I/O call
 - Automatically generate a mapping between the local data buffers and their file layouts
 - Built on top of MPI-IO for parallel file access and high performance
 - Users are responsible to pick the right collective or independent I/O mode

Recent optimizations in MPI-IO

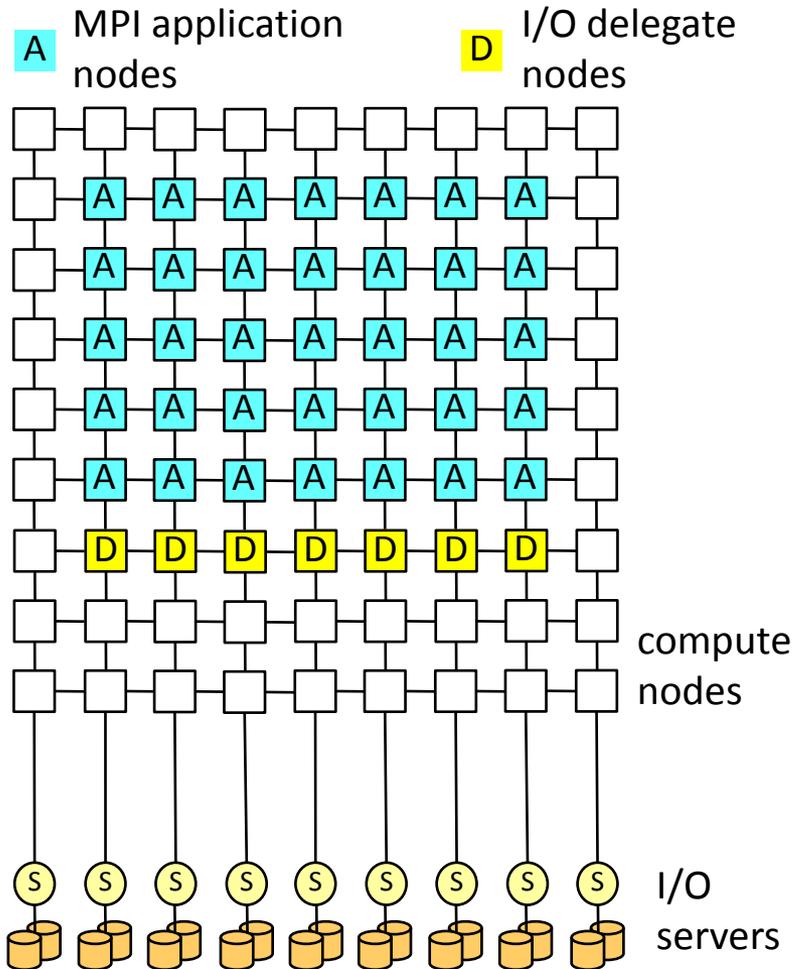
-- file domain alignments --



- An aggregate access region is divided into file domains, one for each I/O process
- Aligned with file stripe (lock) boundaries proves the best performance on parallel file systems
- We developed several alignment methods
 - Static, group-static, neighbor-stripe alignment
 - Demonstrate significant and scalable improvement on Lustre and GPFS

Recent optimizations in MPI-IO

-- I/O delegation --



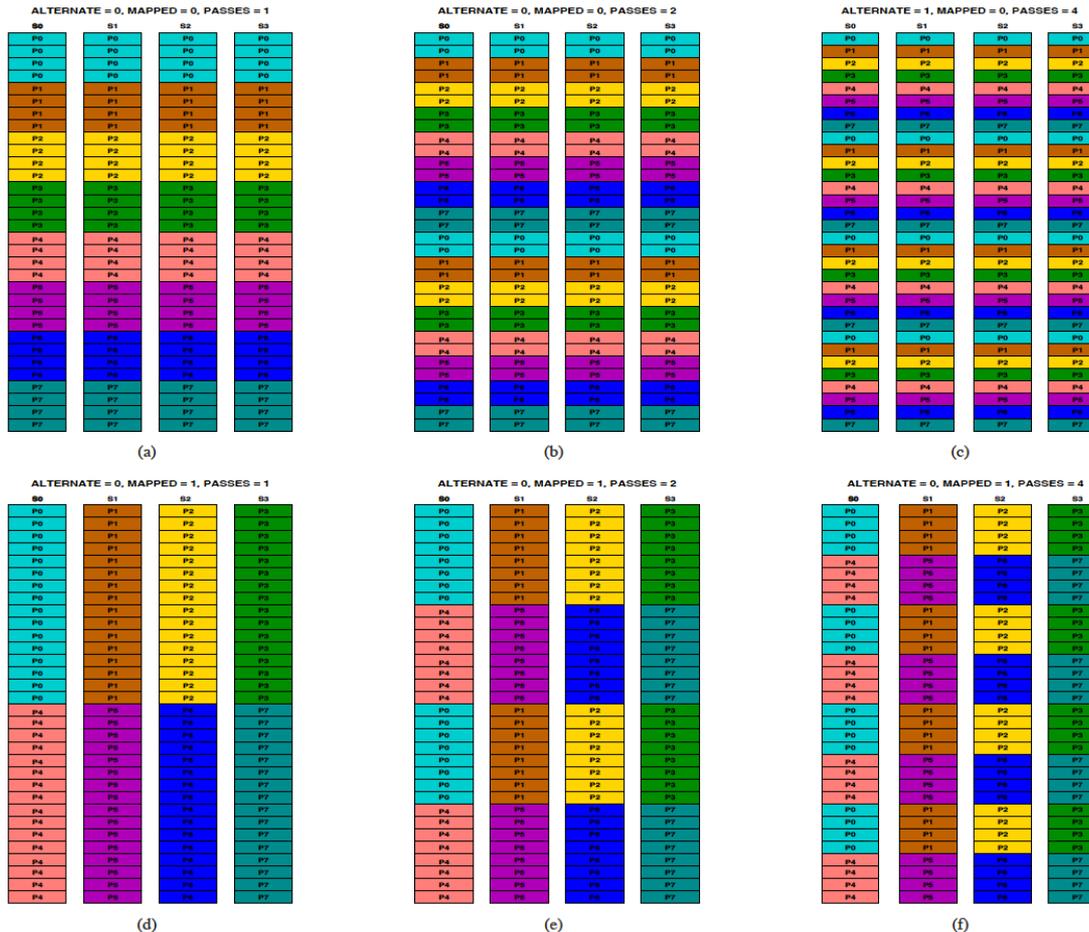
- Additional set of processes
 - Delegates I/O to those processes
 - A layer at the bottom of MPI-IO
 - Uses MPI dynamic process management or split communicator
- Enables I/O optimizations at user space
 - Caching, prefetching, access alignment
- Optimization for independent I/O
 - Important where collective I/O is prohibited

A new I/O benchmark (motivations)

- What is already available at the client side
 - I/O optimizations: aggregation, two-phase I/O, delegation
 - File domain partitioning methods
 - Evaluations show file domain partitioning must match file striping configuration at the server side
- What is available at the server-side
 - Lustre, PVFS allow user-configurable striping setups
 - No single striping configuration can serve well for all kinds of I/O patterns

Access layout mapping options

- Client's access pattern and file layouts at servers

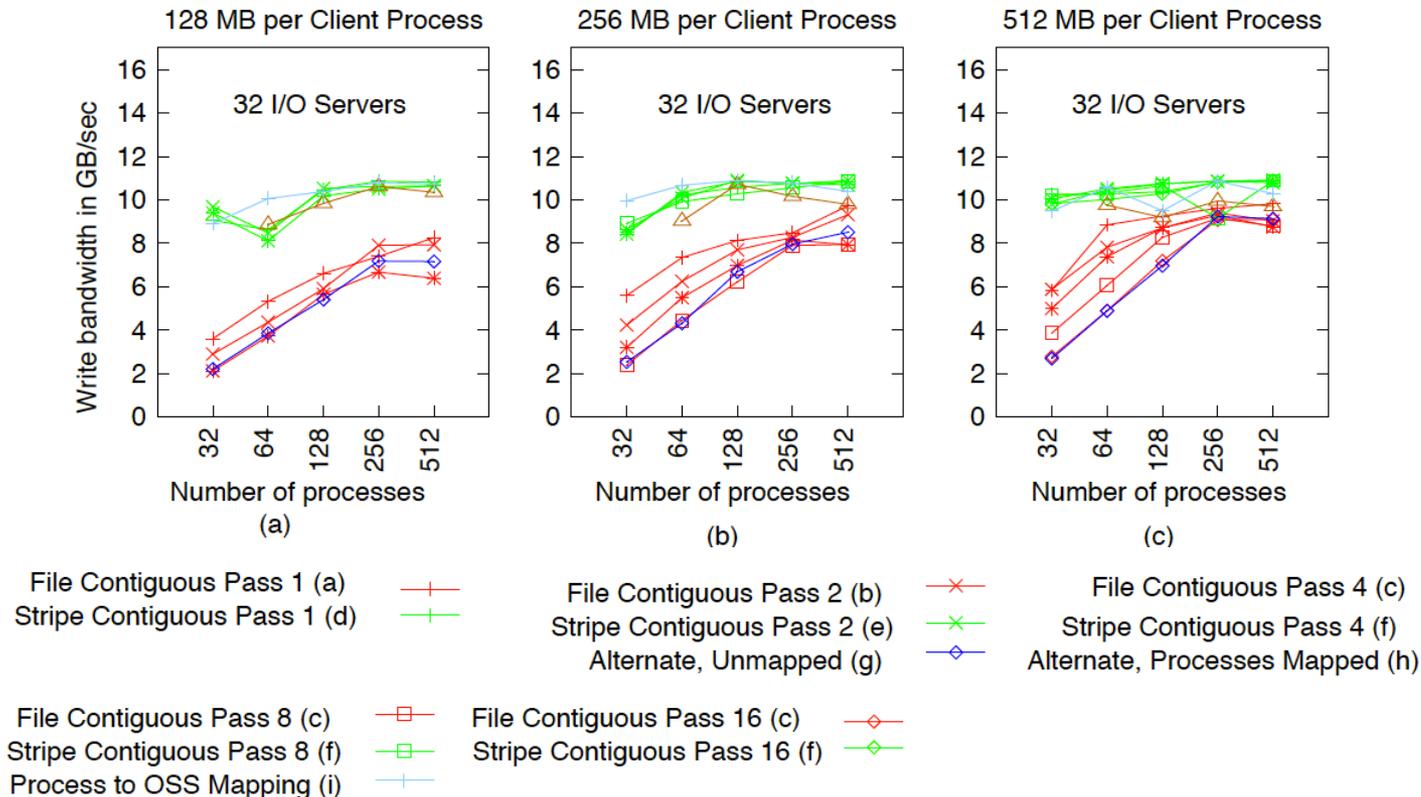


User configurable options

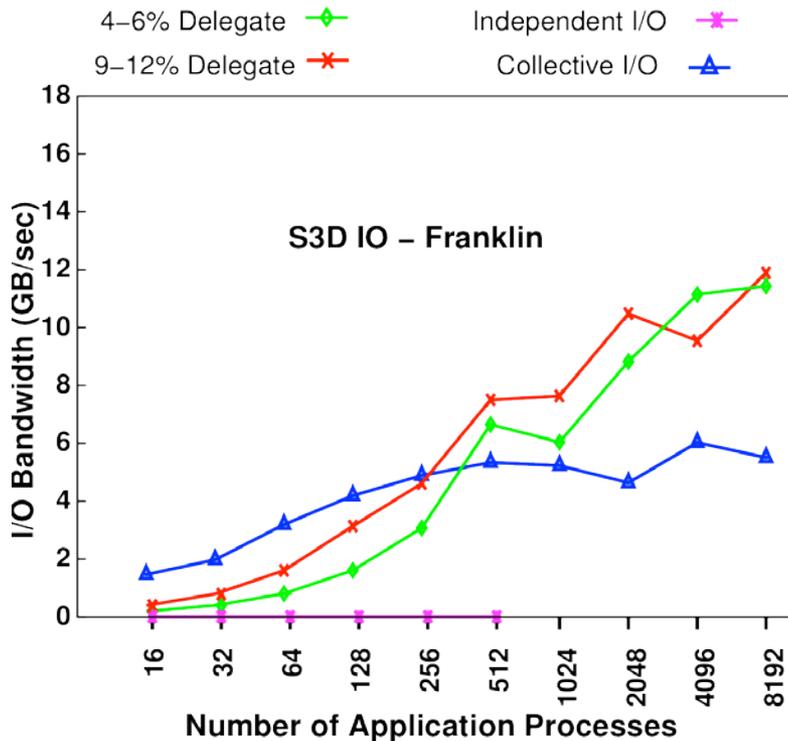
- Given P I/O processes, S file servers, N files
 - Clients' file views
 - Contiguous in file byte-region views
 - Contiguous in server stripe views
 - Load balancing
 - Balance file domain size among I/O processes
 - Balance file server load
 - Number of I/O processes sharing a file
 - Spread N files across S servers
 - Shared-file I/O causes file locking overhead

Benchmark results

- Lustre file system on Franklin, Cray XT4 at NERSC
 - Stripe-contiguous is generally better than byte-contiguous

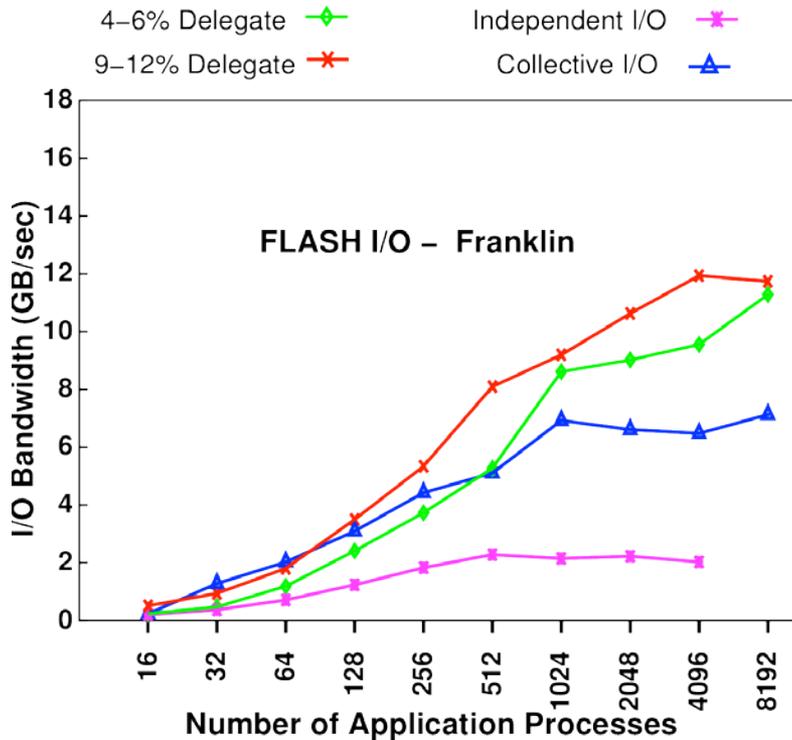


S3D I/O kernels



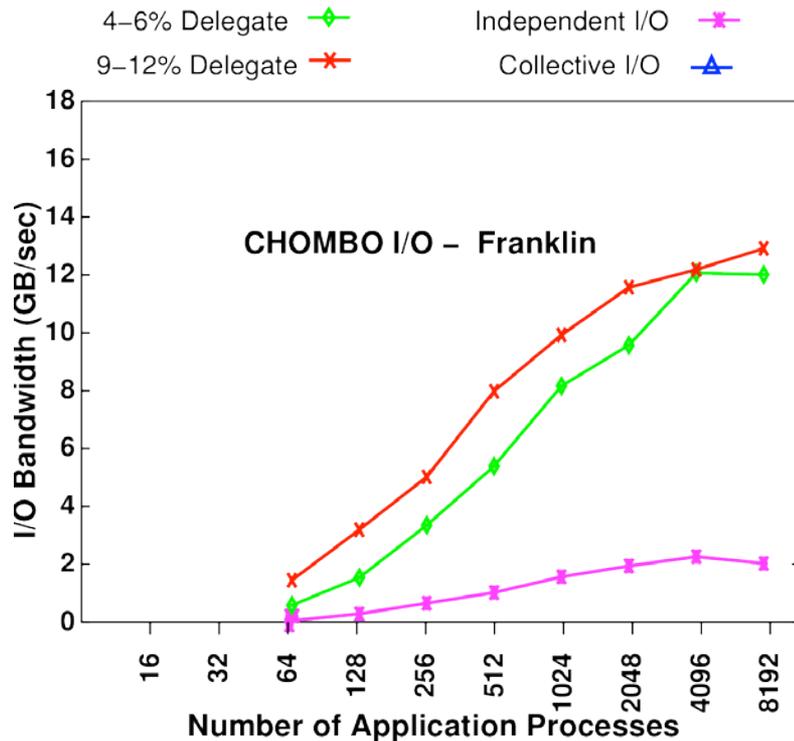
- A combustion computer simulation application developed at Sandia National Lab
 - Block-based data partitioning
- Run with I/O delegation
 - File domain using stripe contiguous method
 - A small percentage of additional MPI processes as I/O delegates

FLASH I/O kernels



- Adaptive Mesh Hydrodynamics simulation developed at U. of Chicago
 - Block-based I/O patterns
 - Balanced I/O load
 - Contiguous access

Chombo I/O



- A software package developed at LBNL for parallel calculations over block-structured, adaptively refined grids
 - Irregular data partitioning
 - Non-contiguous access

Summary

- High-level I/O libraries rely on MPI-IO for high performance
 - MPI-IO optimizations rely on file access domain partitioning methods
 - Example for collective I/O: two-phase I/O method
 - Example for independent I/O: I/O delegation method
 - Optimal file domain partitioning is file system dependent
- A new I/O benchmark
 - Help high-level I/O libraries to choose file striping methods
 - Help MPI-IO to choose file domain partitioning methods
 - An ongoing work