Argonne NATIONAL LABORATORY

… for a brighter future

U.S. Department of Energy

THE UNIVERSITY OF CHICAGO

Office of Science U.S. DEPARTMENT OF ENERGY

A U.S. Department of Energy laboratory managed by The University of Chicago

High-Performance Parallel Data and Storage Management

Rob Latham, Robert Ross, Rajeev Thakur

Argonne National Laboratory

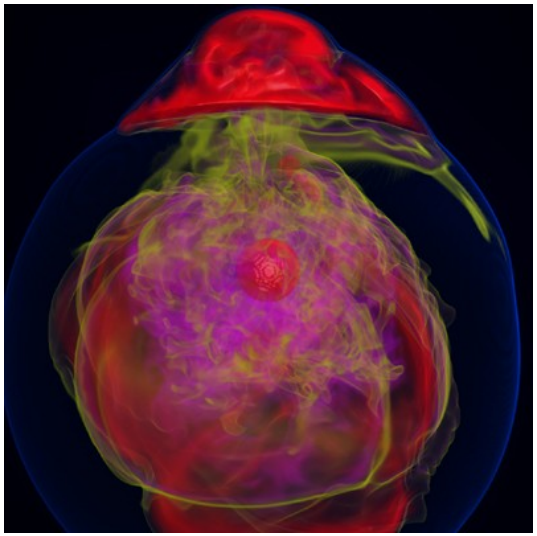
Alok Choudhary

Northwestern University

Computational Science

- Use of computer simulation as a tool for greater understanding of the real world
- Complements experimentation and theory
- As our simulations become ever more complicated
  - Large parallel machines needed to perform calculations
  - Leveraging parallelism becomes more important
- Managing code complexity bigger issue as well
  - Use of libraries increases (e.g. MPI, BLAS)
- Data access is a huge challenge
  - Using parallelism to obtain performance
  - Providing usable and efficient interfaces

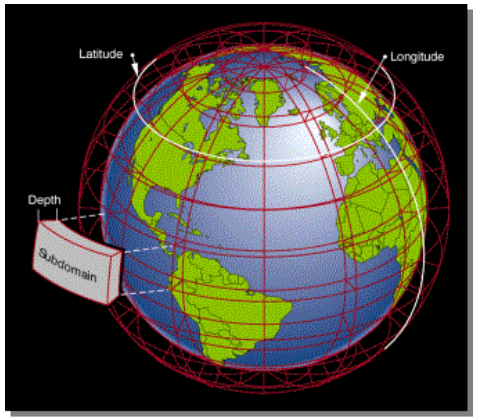
IBM BG/L system.



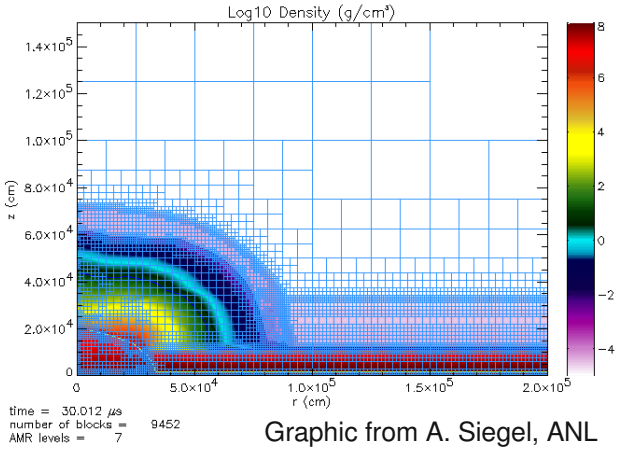
Visualization of entropy in Terascale Supernova Initiative application. Image from Kwan-Liu Ma's visualization team at UC Davis.

Application I/O

- Applications have data models appropriate to domain
  - Multidimensional typed arrays, images composed of scan lines, variable length records
  - Headers, attributes on data
- I/O system as a whole must:
  - **Provide mapping of application data into storage abstractions**
  - **Coordinate access by many processes**
  - **Organize I/O devices into a single space**
- And also
  - Insulate applications from I/O system changes
  - Maintain performance!



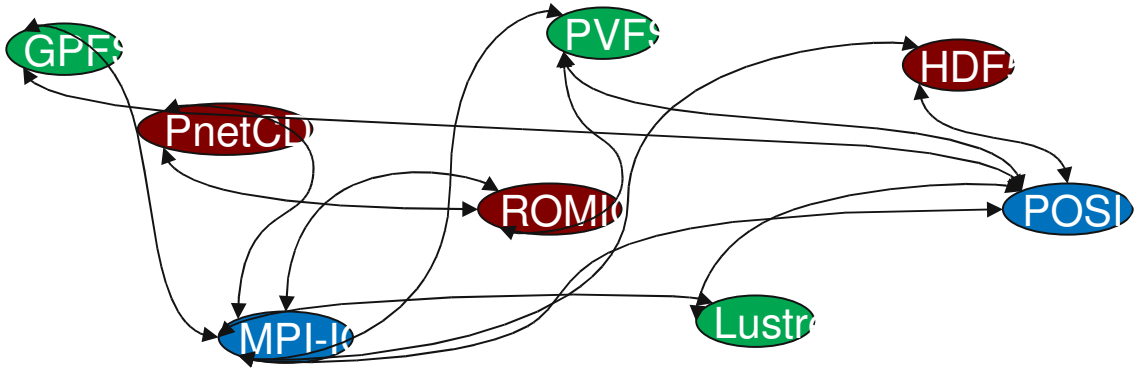
Graphic from J. Tannahill, LLNL



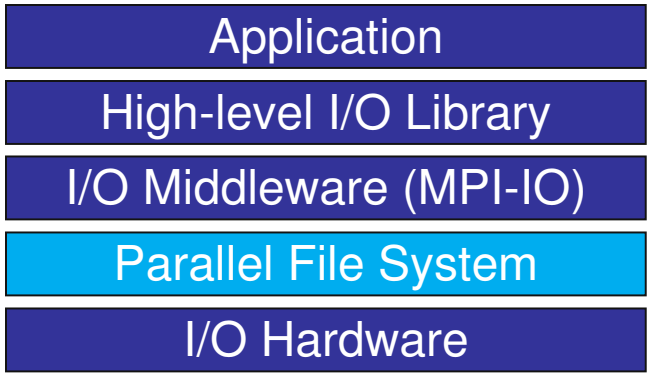
Graphic from A. Siegel, ANL

I/O Tools

- System software and libraries have grown up to address I/O issues
  - Parallel file systems
  - MPI-IO
  - High level libraries
  - Management and data storage
- Relationships between these are not always clear
- Choosing between tools can be difficult

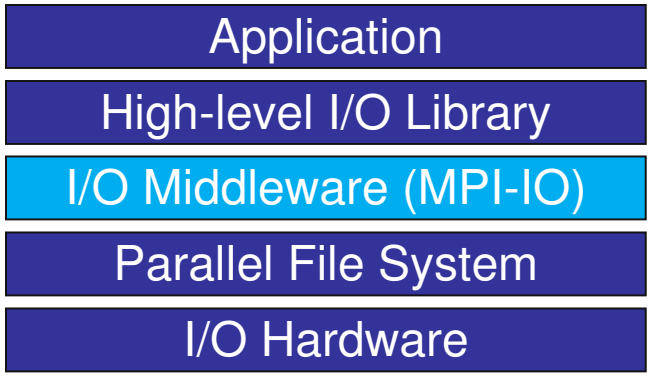


Parallel File System



- Manage storage hardware
  - Present single view
  - Stripe files for performance
- In the context of the I/O software stack
  - Focus on concurrent, independent access
  - Publish an interface that middleware can use effectively
    - *Rich I/O language*
    - *Relaxed but sufficient semantics*
  - Knowledge of collective I/O usually very limited

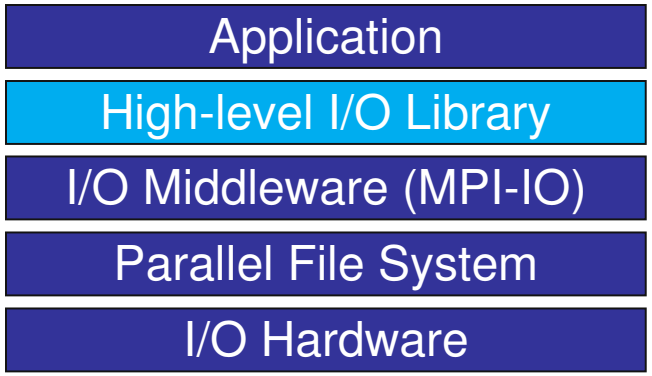
I/O Middleware



- Match the programming model (e.g. MPI)
- Facilitate concurrent access by groups of processes
  - Collective I/O
  - Atomicity rules
- Expose a generic interface
  - Good building block for high-level libraries
- Efficiently map middleware operations into PFS ones
  - Leverage any rich PFS access constructs, such as:
    - *Scalable file name resolution*
    - *Rich I/O descriptions*

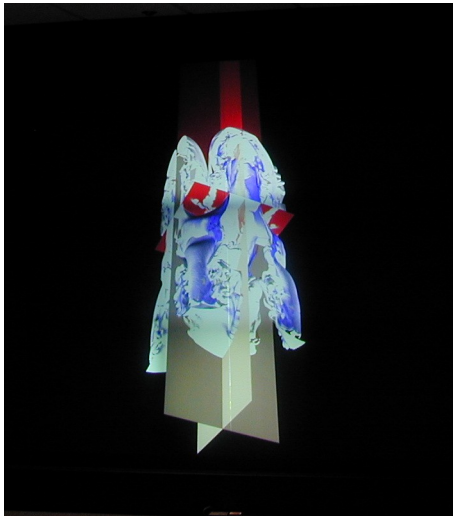
High Level Libraries

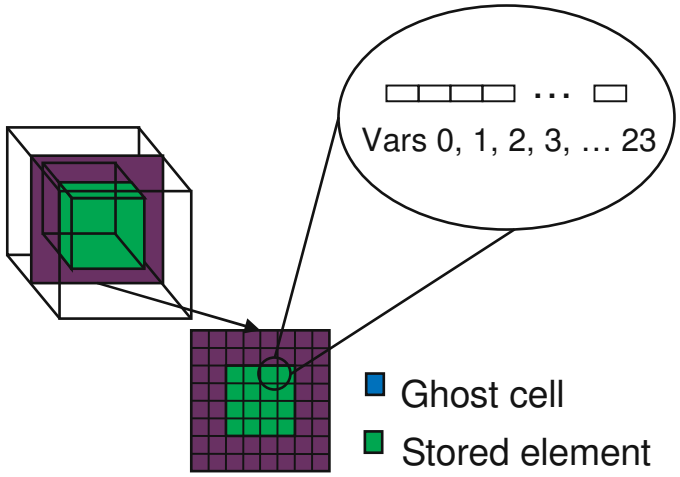
- Match storage abstraction to domain
  - Multidimensional datasets
  - Typed variables
  - Attributes
- Provide self-describing, structured files
- Map to middleware interface
  - Encourage collective I/O
- Implement optimizations that middleware cannot, such as
  - Caching attributes of variables
  - Chunking of datasets



Stack in Action: FLASH Astrophysics

- FLASH is an astrophysics code for studying events such as supernovae
  - Adaptive-mesh hydrodynamics
  - Scales to 1000s of processors
  - MPI for communication
- Frequently checkpoints:
  - Large blocks of typed variables from all processes
  - Portable format
  - Canonical ordering (different than in memory)
  - Skipping ghost cells





FLASH and I/O Stack: Application

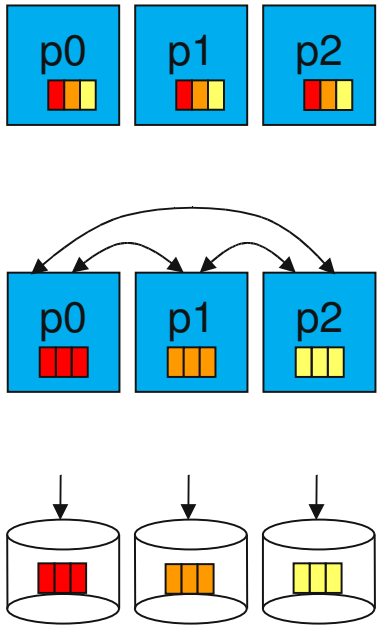
- FLASH performs high level I/O operations
  - Write out multi-dimensional arrays
  - Annotate data
    - *Timestamps, machine information, provenance*
- FLASH developers focus on science
  - Most developer effort on simulation/visualization
  - Data management handled in small "I/O kernel"
    - *Can experiment with different high-level libraries*

FLASH and I/O Stack: High-Level Library

- FLASH can use either Parallel HDF5 or Parallel-NetCDF
- High-level library deals with complexity/power of MPI-IO
  - Computes MPI file views
  - Coordinates collective I/O
  - Sets MPI-IO hints
- Library has additional context for optimizations
  - Parallel-NetCDF exploits *define mode* vs *data mode*
  - Parallel HDF5 has I/O filters, data chunking
- Library also takes care of file format
  - Self describing
  - Platform independent

FLASH and I/O Stack: Middleware

- In FLASH case, middleware is MPI-IO
- MPI-IO standard defines lots of “tuning knobs”
  - Hints, noncontiguous I/O, collective I/O
- MPI-IO implementation optimizes access
  - Data sieving, collective buffering, data shipping
- Deals directly with file system
  - FS-specific APIs, tuning parameters
- *Noncontiguous FLASH request becomes contiguous file system request*



MPI-IO two phase I/O optimization

FLASH and I/O Stack: File system

- FLASH runs on many machines, deals with many parallel file systems
  - But middleware and high-level library make FLASH fs-agnostic
- Parallel File System
  - Manages blocks on disk
  - Namespace
  - Coordinates parallel access
- Advanced file systems might have additional features
  - Object-based storage
  - Highly expressive request language

Research Areas

- High-Level Libraries and widely-understood file format
  - Data Mining with external packages
- Collaborative caching
- Integrated I/O forwarding
- Ongoing research in optimization and enhancements
- High performance parallel data storage
- Improve fit for application workloads
  - Multidimentional arrays common, not exclusive
  - AMR, Distributed Mesh
  - Going to require new libraries: not a “next week” project.
- Always looking for additional I/O kernels
  - Applications benefit from intense study of their specific I/O patterns
  - Developers benefit from broader base of “real world” access patterns

Resources

- Parallel-NetCDF: www.mcs.anl.gov/parallel-netcdf
- Parallel HDF5: http://hdf.ncsa.uiuc.edu/HDF5/PHDF5/
- ROMIO: www.mcs.anl.gov/romio
- PVFS: www.pvfs.org
- SciDAC SDM Center: http://sdm.lbl.gov/sdmcenter/