Parallel netCDF:
A High-Performance Scientific I/O Interface

Jianwei Li

ECE Department, Northwestern University

(jianwei@ece.northwestern.edu)

W. Liao, A. Choudhary (Northwestern University)

R. Ross, R. Thakur, W. Gropp, R. Latham, A. Siegel (Argonne National Laboratory)

B. Gallagher (University of Chicago)

M. Zingale (University of California, Santa Cruz)

November 20, 2003

Outline

- Overview of netCDF & our motivation
- NetCDF file format & current serial interface
- Access netCDF files in parallel applications
- Parallel netCDF design & implementation
- Parallel netCDF vs Parallel HDF5
- Performance evaluation
- Summary & future work
- Software and documentation resources

Introduction

- netCDF (network Common Data Form)
  - Initially by Unidata Program Center in Boulder, Colorado.
  - Provide a common data access method for various scientific apps.
- File format & programming interface
  - Portable, platform independent and self-describing file format
  - Libraries for array-oriented data access in C, FORTRAN, etc.
- A broad community of netCDF users
  - Easy to learn and use
  - A de facto data access standard for much of the climate community
  - atmospheric app., ocean modeling, fusion simulations, medical images, satellite/radar images
  - Huge amount of existing netCDF datasets, and processing tools
- Well supported by Unidata
- Parallel interface support from NWU/ANL

Why Parallel I/O?

- Most scientific applications are in parallel
- Increasing requirement on data amount
- I/O bottleneck
- Scalability needs parallel I/O
- Programming convenience with parallel I/O
- Significant I/O performance improvement with appropriate parallel I/O techniques
- Well studied and supported

NetCDF File Format

header:

definition of the netCDF dataset

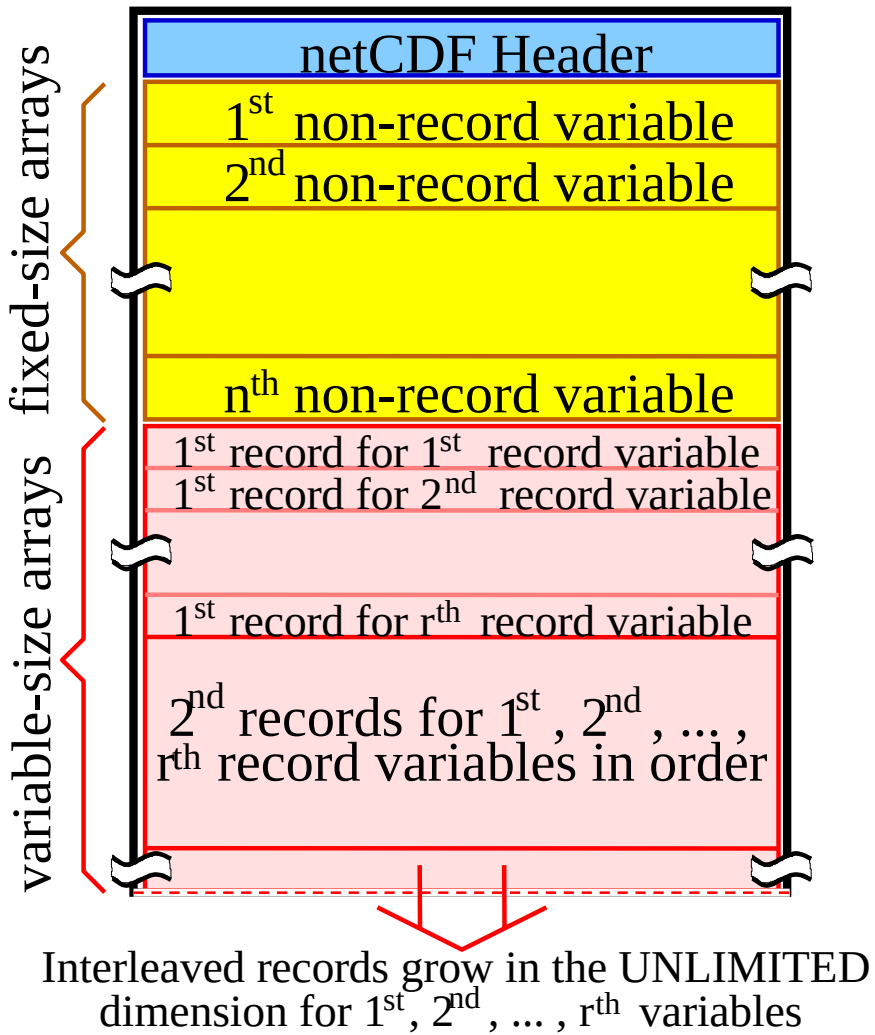
n fixed-size variables:

arrays of fixed dimensions

r record variables:

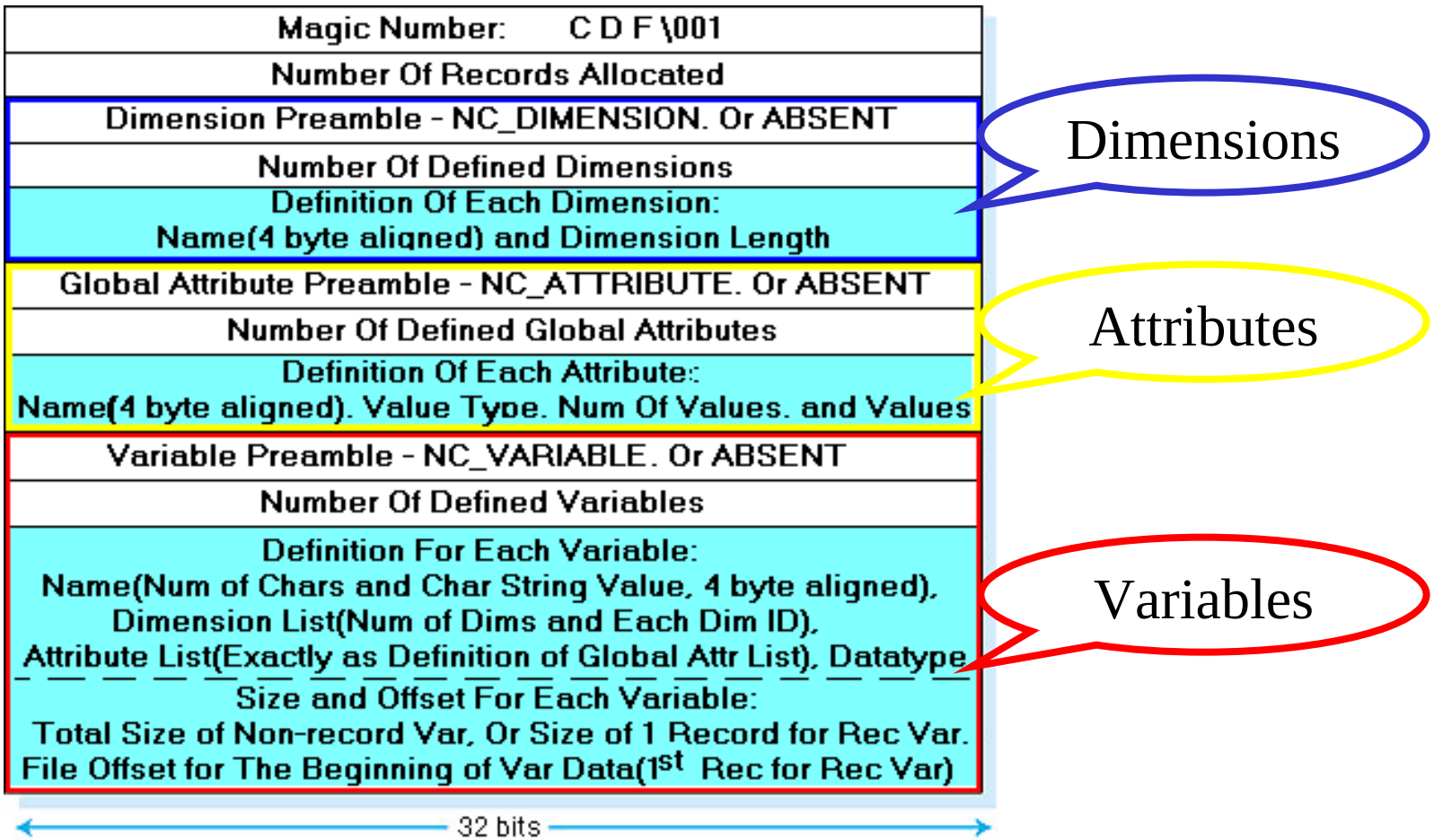
arrays with its most-significant dimension UNLIMITED,

records defined by the rest of the dimensions.



File Header

Defines the 3 components of a netCDF dataset:



An Example NetCDF File

```
netCDF_example { // CDL notation for netCDF dataset
   dimensions: // dimension (names and lengths)
        lat = 5, lon = 10, level = 4, time = unlimited;
   variables: // var (types, names, shapes, attributes)
        float temp(time,level,lat,lon);
                temp:long_name = "temperature";
                temp:units = "celsius";
        float rh(time,lat,lon);
                rh:long_name = "relative humidity";
                rh:valid_range = 0.0, 1.0; // min and max
   // global attributes:
        :source = "Fictional Model Output";
   data: // optional data assignments
        temp =  0, ..., 0;
        rh   =  .5,.2,.4,.2,.3,.2,.4,.5,.6,.7,
                .1,.3,.1,.1,.1,.1,.5,.7,.8,.8,
                .1,.2,.2,.2,.2,.5,.7,.8,.9,.9,
                .1,.2,.3,.3,.3,.3,.7,.8,.9,.9,
                 0,.1,.2,.4,.4,.4,.4,.7,.9,.9;  // 1 record allocated
}
```

Serial NetCDF API

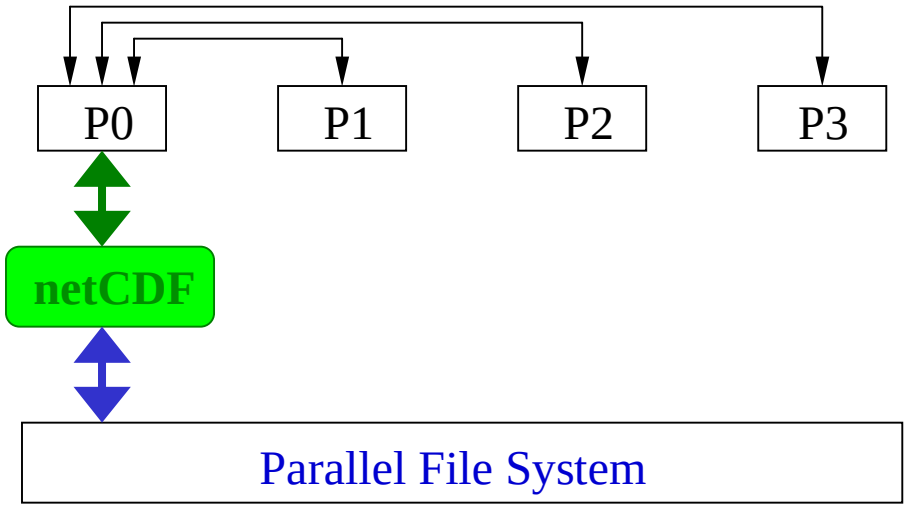
- **Dataset Functions**
  - create/open/close/abort a dataset
  - set the dataset to define/data mode
  - synchronize dataset changes to storage
- **Define Mode Functions**
  - define dataset dimensions and variables
- **Attribute Functions**
  - manage adding, changing, and reading attributes of the dataset
- **Inquiry Functions**
  - return dataset metadata:

    dim(id, name, len), var(id, name, ndims, shape, etype), # of dims/vars/attributes, etc.
- **Data Access Functions**
  - read/write variable data in one of the five access methods (patterns):

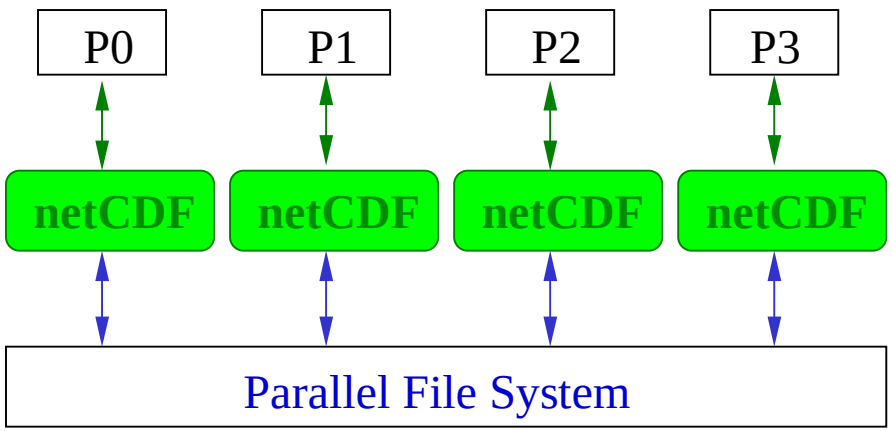
    single element, whole array, subarray, strided subarray, or mapped strided subarray

Built on native I/O and designed for serial access

Parallel Access to NetCDF Files?

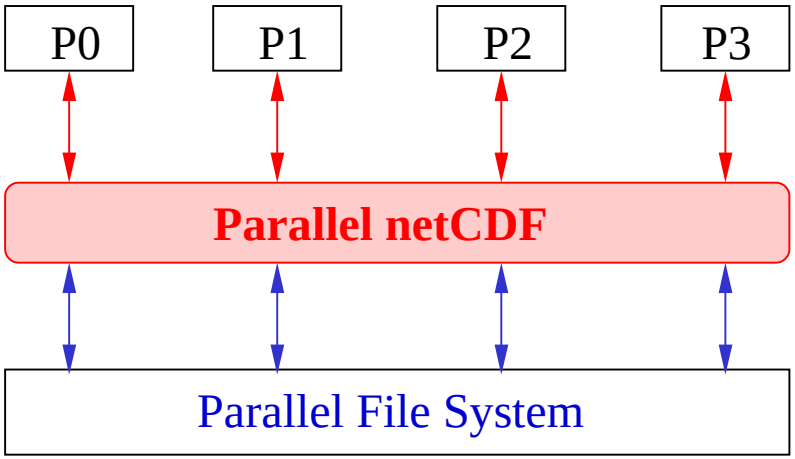


- Slow and cumbersome
- Data shipping
- I/O bottleneck
- Memory requirement



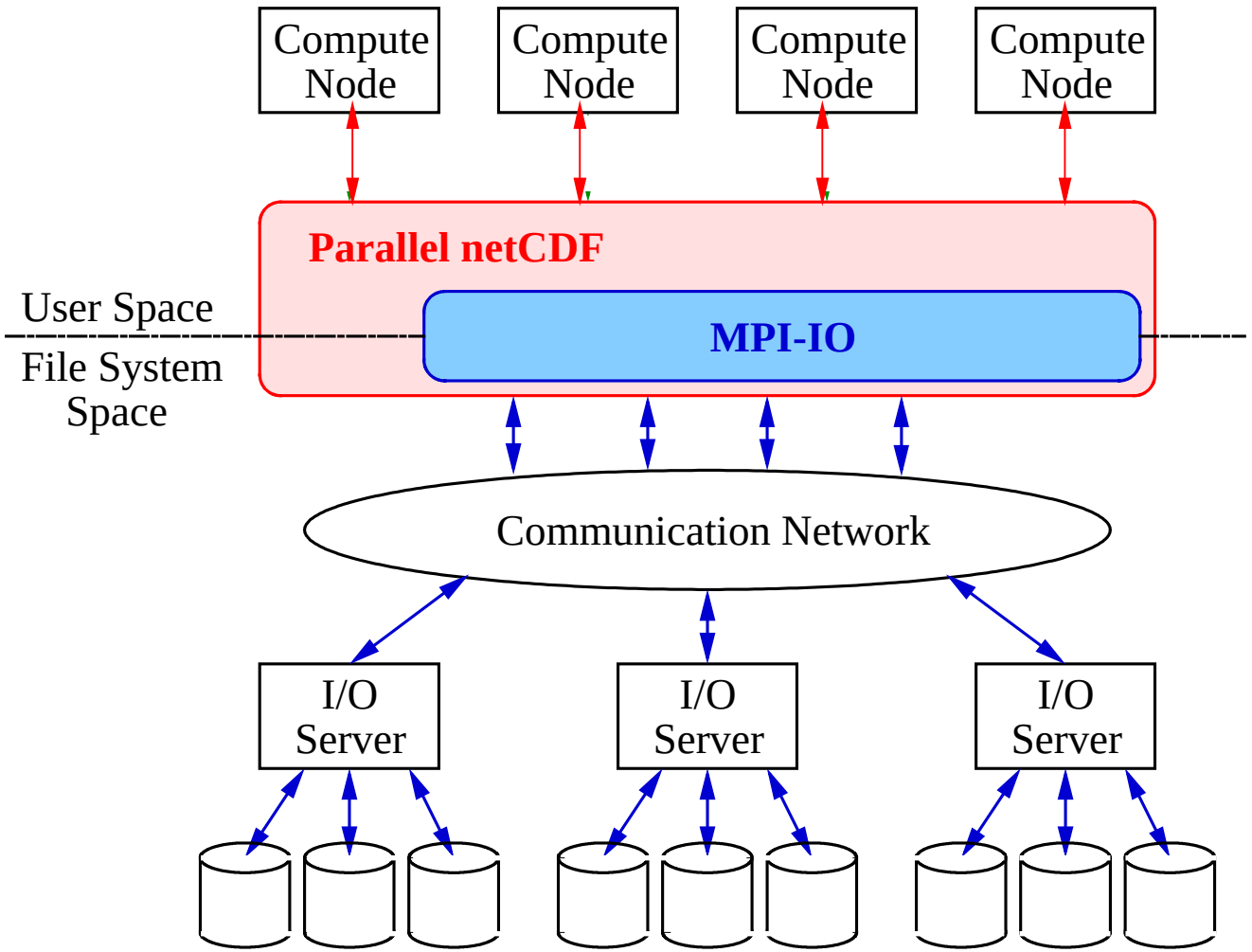
- Multiple file pieces
- Breaks the netCDF dataset
- Independent Access

Access through Parallel NetCDF!



- New parallel interface
- Perform I/O cooperatively or collectively
- Potential parallel I/O optimizations for better performance
- NetCDF data integration and management

Architecture for Parallel NetCDF



Parallel NetCDF API

- The same netCDF file format
- Maintains the look and feel of the serial netCDF API
  - Same syntax & semantics for:
    - Dataset functions (except create/open)
    - Define mode functions, attribute functions and inquiry functions
    - High-level data access functions
  - Distinguish by prefixing function calls with “ncmpi_”/“nfmpi_”
- Parallel access through MPI-IO
  - Benefit from MPI-IO optimizations (data shipping, two-phase I/O, etc.)
  - MPI communicator added in the argument list for create/open
  - MPI_Info used for parallel I/O management and further optimization
  - Collective I/O (function names end with “_all”) vs noncollective I/O
- New flexible data access functions
  - Accept non-contiguous memory regions described by MPI datatypes

PnetCDF Example Code

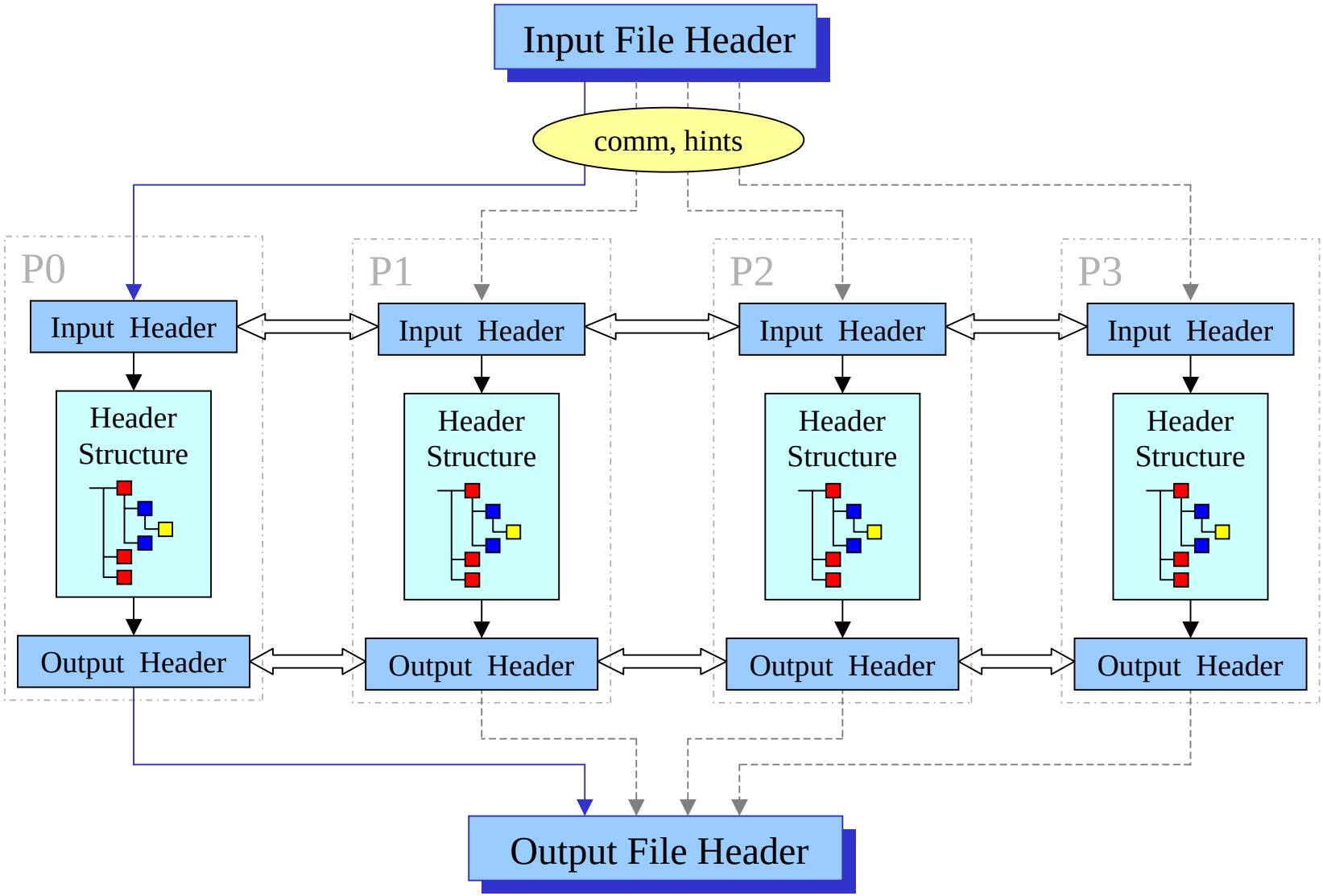
File write

```
1. ncmpi_create(mpi_comm, filename, open_mode, mpi_info, &file_id);
2. ncmpi_def_var(file_id, ...);
   . . .      . . .
   ncmpi_enddef(file_id);
3. ncmpi_put_vara_all(file_id, var_id, start[], count[], buffer, bufcount, mpi_datatype);
4. ncmpi_close(file_id);
```

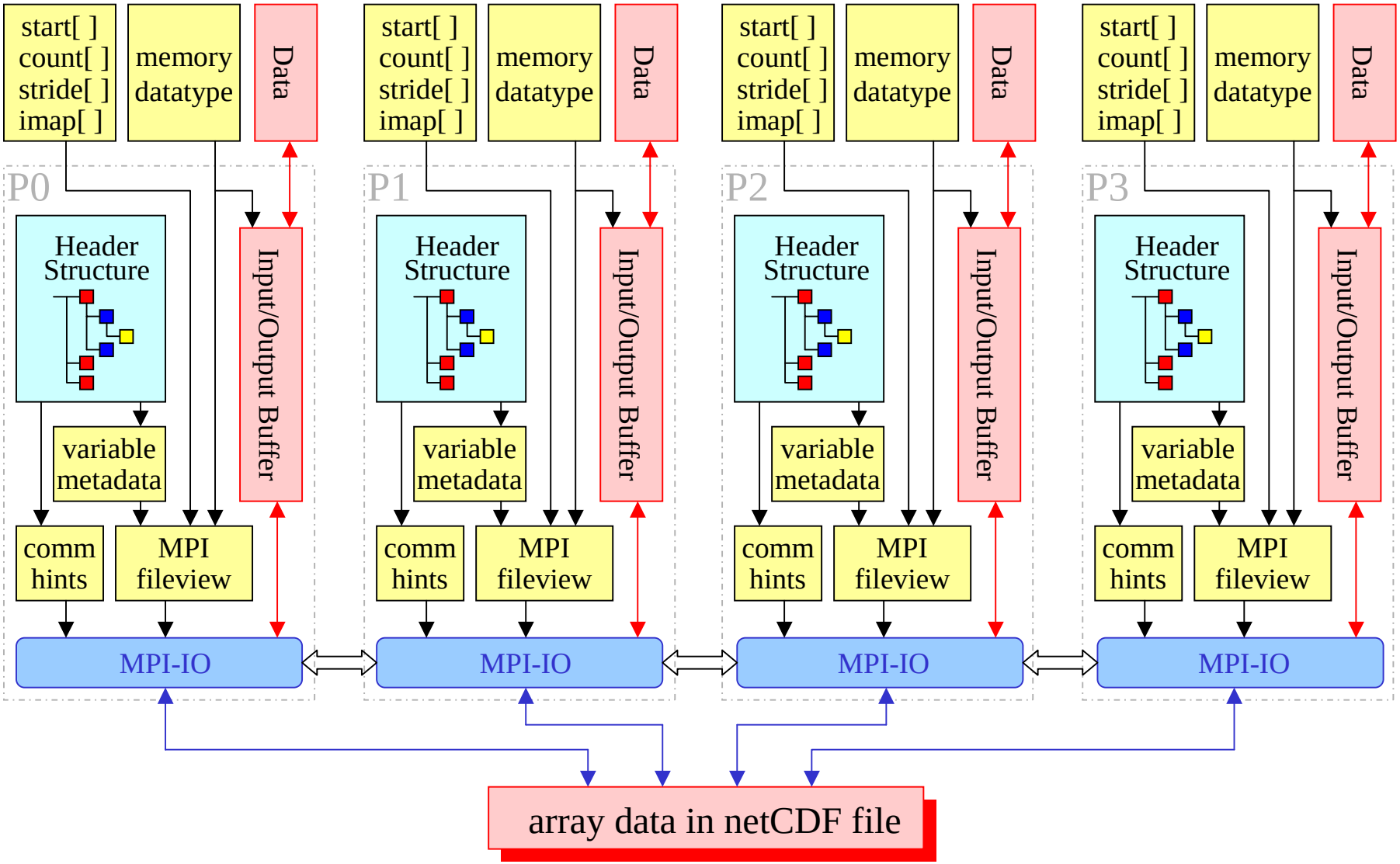
File read

```
1. ncmpi_open(mpi_comm, filename, open_mode, mpi_info, &file_id);
2. ncmpi_inq(file_id, ... );
   . . .      . . .
3. ncmpi_get_vars(file_id, var_id, start[], count[], stride[], buffer, bufcount, mpi_datatype);
4. ncmpi_close(file_id);
```

Access to File Header

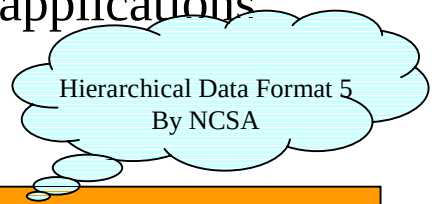


Parallel I/O for Array Data



Compare with Parallel HDF5

- Both define portable, self-describing file formats
- Both serve as data access standards for parallel scientific applications
- Both have parallel I/O built on top of MPI-IO

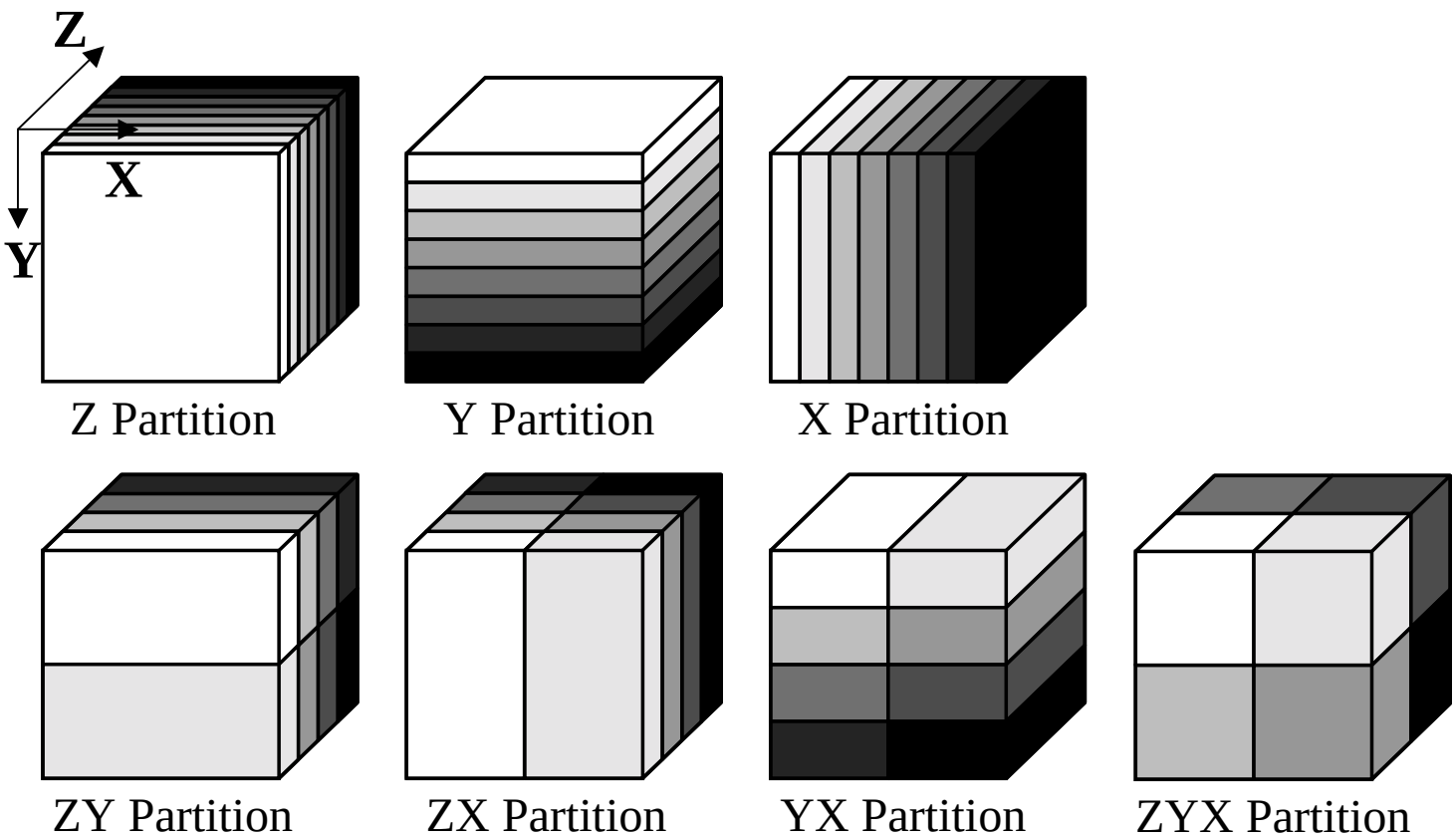


Parallel netCDF	Parallel HDF5
• Contiguous or interleaved data layout; • Sits on top of MPI-IO, transferring user access patterns directly to MPI fileview, little overhead.	• Tree-like file structure; • Use dataspace and hyperslabs to define data organization, map and transfer data between memory and filespace, pack or unpack.
• One time header I/O gets all necessary info for direct access of each data array; • By maintaining a local copy of header, each process can access any array identified by its permanent ID at any time, without any collective open/close operation of the object. Saves a lot of inter-process communication.	• Iterate through entire namespace to get the header info to access each object; • Need collective open/close operation when accessing each single object, and for file write, the metadata need to be updated in a synchronous way. So lots of communication overhead.

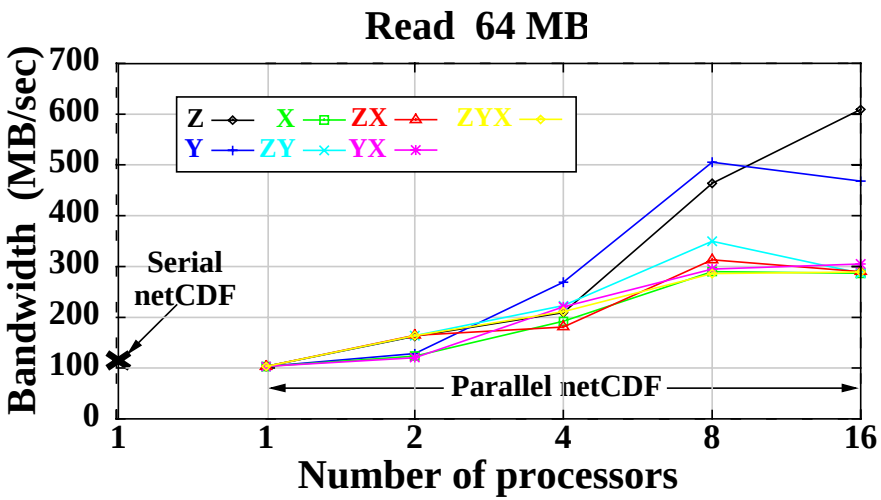
Performance Evaluation

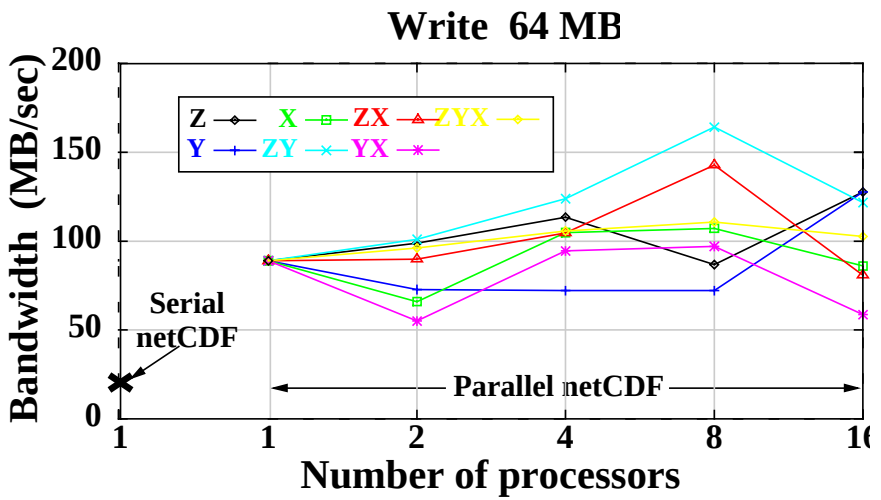
- A micro-benchmark accessing a 3-D array
  - Platform: Bluehorizon (IBM SP-2) @ SDSC
    - IBM's MPI, GPFS file system
  - Performance scalability of our PnetCDF
  - Compare Parallel netCDF with Serial netCDF
- FLASH I/O benchmark writing a number of multi-dimensional arrays
  - Platform: ASCI White (Frost) @ LLNL
    - IBM's MPI, GPFS file system
  - Compare PnetCDF with PHDF5

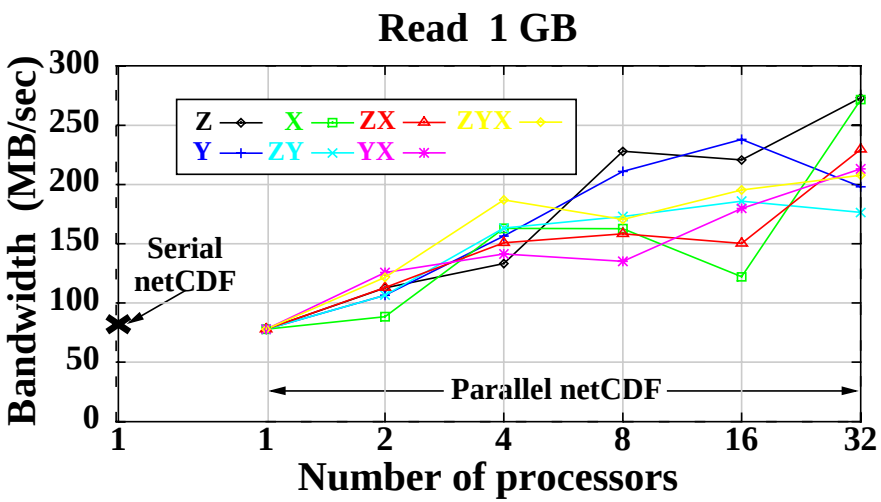
Data Partition of Micro-benchmark

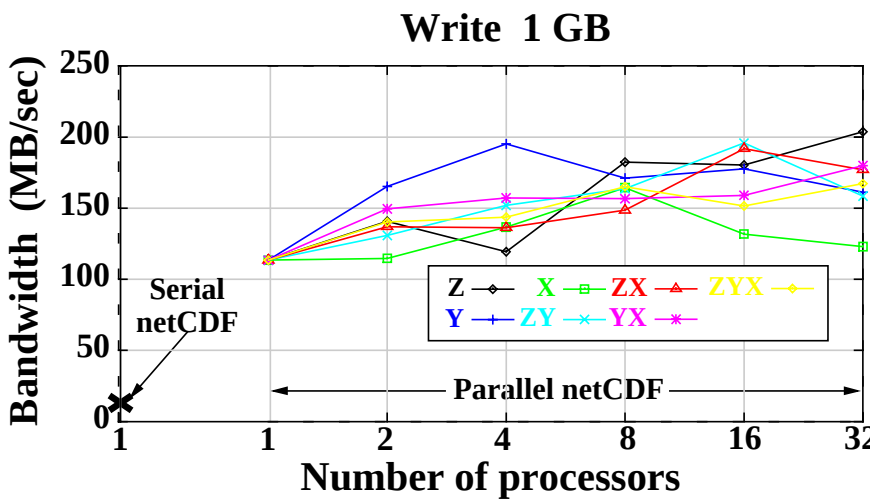


Performance Scalability





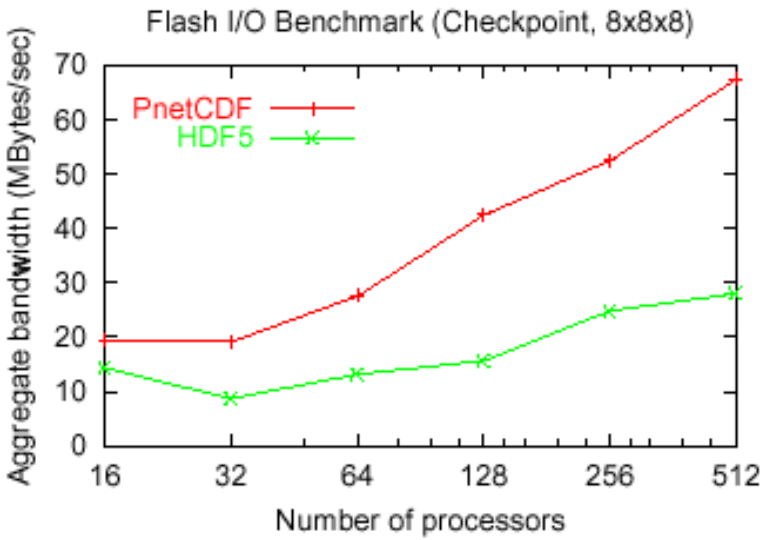


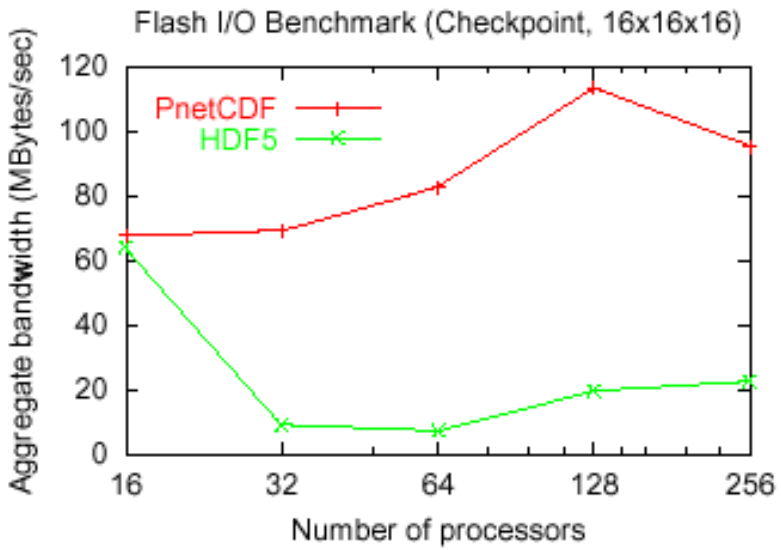


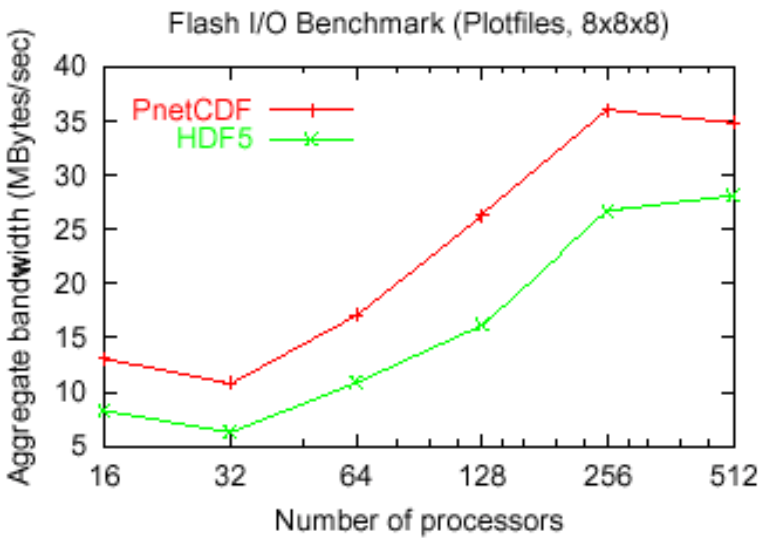
FLASH I/O Benchmark

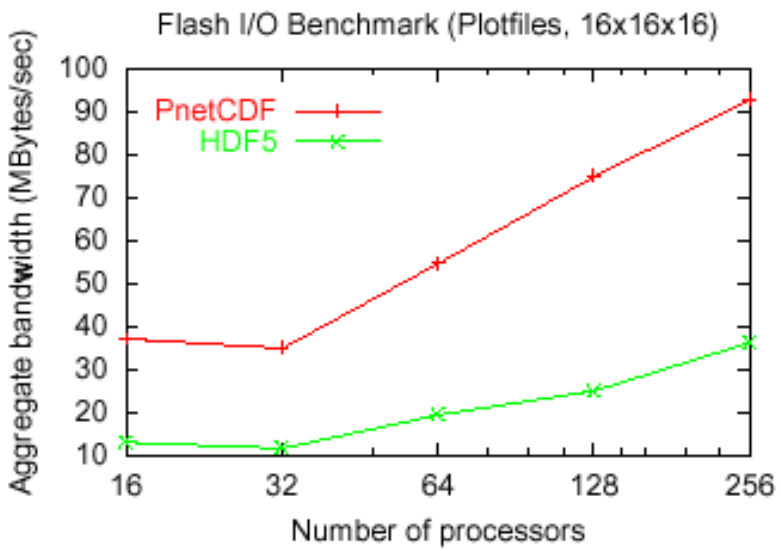
- FLASH – parallel hydrodynamics code
  - Simulate astrophysical thermonuclear flashes in 2/3D
  - Use structured adaptive mesh refinement method to solve some physics equations
  - Produce checkpoint files and visualization plotfiles
- FLASH I/O benchmark
  - Simulate the I/O pattern of FLASH
  - 3D AMR subblocks with a perimeter of four guard cells (In the simulation, 80 blocks by each process, 8*8*8 or 16*16*16)
  - Generate a checkpoint file, a plotfile with centered data, and a plotfile with corner data
  - A series of multi-dimensional arrays, each partitioned in the most significant dimension

FLASH I/O Performance

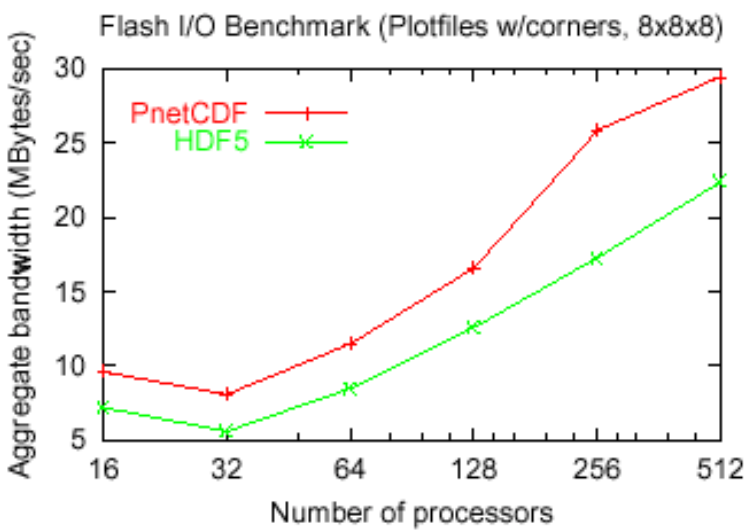


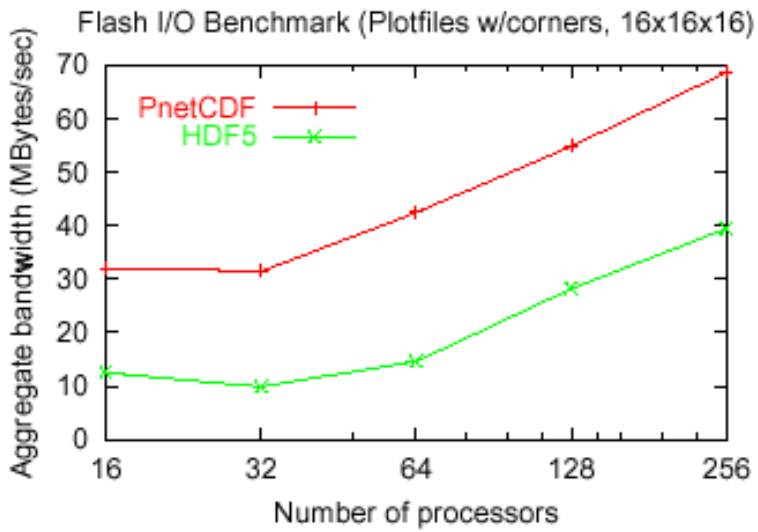






FLASH I/O Performance (cont.)





Summary and Future Work

- PnetCDF - A new high-level I/O library
  - Uses same data format as netCDF
  - Maintains the flavor of the netCDF API
  - Provides parallel access semantics
- Current status
  - Distributions of code available, w/ basic test suite
  - Mailing list for users, support
  - External groups are already adopting this software
  - Performance is already competitive w/out any tuning
- Future work
  - Perfect the library, improving our test suite, support
  - Performance tuning for key platforms (IBM SP, Linux)

More Information

- Learn more about netCDF

  http://www.unidata.ucar.edu/packages/netcdf/
- Download the PnetCDF software & docs

  http://www.mcs.anl.gov/parallel-netcdf/

Thank you!

>> Questions???