

Parallel I/O Performance Study and Optimizations with HDF5, A Scientific Data Package

MuQun Yang, Christian Chilan, Albert Cheng, Quincey Koziol, Mike Folk, Leon Arber

> The HDF Group Champaign, IL 61820

Outline

- Introduction to parallel I/O library
- HDF5,netCDF,netCDF4
- Parallel HDF5 and Parallel netCDF performance comparison
- Parallel netCDF4 and Parallel netCDF performance comparison
- Collective I/O optimizations inside HDF5
- Conclusion

I/O process for Parallel Application



- P0 becomes bottleneck
- May exceed system memory



- May achieve good performance
- Needs post-processing(?)
- More work for applications

I/O process for Parallel Application



HDF5 VS netCDF

- HDF5 and netCDF provide data formats and programming interfaces
- HDF5
 - Hierarchical file structures
 - Flexible data model
 - Many Features
 - In-memory compression filters
 - Chunked storage
 - Parallel I/O through MPI-IO

NetCDF

- Linear data layout
- A parallel version of netCDF from ANL/Northwestern U. (PnetCDF) provide support for parallel access on top of MPI-IO

Overview of NetCDF4

Advantages:

- New features provided by HDF5:
 - □ More than one unlimited dimension
 - □ Various compression filters
 - Complicate data type such as struct or array datatype
 - □ Parallel IO through MPI-IO
- NetCDF-user friendly APIs
- Long-term maintenance and distribution
- Potential larger user community

Disadvantage:

Install HDF5 library

An example for collective I/O

- Every processor has a noncontiguous selection.
- Access requests are interleaved.
- Write operation with 32 processors, each processor selection has 512K rows and 8 columns (32 MB/proc.)

□ Independent I/O: 1,659.48 s. P0 P1 P2 P3

□ Collective I/O: 4.33 s.



Outline

- Introduction to parallel I/O library
- HDF5,netCDF,netCDF4
- Parallel HDF5 and Parallel netCDF performance comparison
- Parallel netCDF4 and Parallel netCDF performance comparison
- Collective I/O optimizations inside HDF5
- Conclusion

Parallel HDF5 and PnetCDF performance comparison

- Previous Study:
 - PnetCDF claims higher performance than HDF5
- NCAR Bluesky
 - Power4

LLNL uP

Power5

PnetCDF 1.0.1 vs. HDF5 1.6.5.

HDF5 and PnetCDF performance comparison

- Benchmark is the I/O kernel of FLASH.
- FLASH I/O generates 3D blocks of size 8x8x8 on Bluesky and 16x16x16 on uP.
- Each processor handles 80 blocks and writes them into 3 output files.
- The performance metric given by FLASH I/O is the parallel execution time.
- The more processors, the larger the problem size.

Previous HDF5 and PnetCDF Performance Comparison at ASCI White

Flash I/O Benchmark (Checkpoint Files)



HDF5 and PnetCDF performance comparison

Flash I/O Benchmark (Checkpoint files) Flash I/O Benchmark (Checkpoint files) PnetCDF — HDF5 independent PnetCDF — HDF5 independent MB/s MB/s Number of Processors Number of Processors

Bluesky: Power 4

uP: Power 5

HDF5 and PnetCDF performance comparison

Flash I/O Benchmark (Checkpoint files)



Flash I/O Benchmark (Checkpoint files)

ROMS

- Regional Oceanographic Modeling System
- Supports MPI and OpenMP
- I/O in NetCDF
- History file writer in parallel

Data:

□ 60 1D-4D double-precision float and integer arrays

PnetCDF4 and PnetCDF performance comparison



- Fixed problem size = 995 MB
- Performance of PnetCDF4 is close to PnetCDF

ROMS Output with Parallel NetCDF4



- The IO performance gets improved as the file size increases.
- It can provide decent I/O performance for big problem size.

Outline

- Introduction to parallel I/O library
- HDF5,netCDF,netCDF4
- Parallel HDF5 and Parallel netCDF performance comparison
- Parallel netCDF4 and Parallel netCDF performance comparison
- Collective I/O optimizations inside HDF5
- Conclusion

Improvements of collective IO supports inside HDF5

- Advanced HDF5 feature: non-regular selections
- Performance optimizations: chunked storage
 - Provide several IO options to achieve good collective IO performance
 - Provide APIs for applications to participate in the optimization process

Improvement 1 HDF5 non-regular selections

2-D array with the IO in shaded selections



- Only one HDF5 IO call
- Good for collective IO

HDF5 chunked storage



- Required for extendable data variables
- Required for filters
- Better subsetting access time

For more information about chunking: http://hdf.ncsa.uiuc.edu/UG41r3_html/Perform.fm2.html#149138

Performance issue:

Severe performance penalties with many small chunk IOs

Improvement 2: One linked chunk IO



One MPI Collective IO call

Improvement 3: Multi-chunk IO Optimization

- Have to keep the option to do collective IO per chunk
 Collective IO bugs inside different MPI-IO packages
 Limitation of system memory
- Problem

□ Bad performance caused by improper use of collective IO



Just use independent IO

Improvement 4

Problem

HDF5 may not have enough information to make the correct decision about the way to do collective IO

Solution

Provide APIs for applications to participate in the decision-making process Flow chart of Collective Chunking IO improvements inside HDF5



For the detailed about performance study and optimization inside HDF5:

http://hdf.ncsa.uiuc.edu/HDF5/papers/papers/ParallelIO/ParallelPerformance.pdf

http://hdf.ncsa.uiuc.edu/HDF5/papers/papers/ParallelIO/HDF5-CollectiveChunkIO.pdf

Conclusions

- HDF5 provides collective IO supports for non-regular selections
- Supporting collective IO for chunked storage is not trivial. Users can participate in the decision-making process that selects different IO options.
- I/O Performance is quite comparable when parallel NetCDF and parallel HDF5 libraries are used in similar manners.
- I/O performance of parallel NetCDF4 is compatible with parallel NetCDF with about 15% slowness in average for the output of ROMS history file. We suspect that the slowness is due to the software management when passing information from parallel NetCDF4 to HDF5.

Acknowledgments

This work is funded by National Science Foundation Teragrid grants, the Department of Energy's ASC Program, the DOE SciDAC program, NCSA, and NASA.