Performance Study of Parallel NetCDF4 in ROMS

MuQun Yang
June 30th, 2006
Revised,
February 7th, 2007

Abstract

This report presents IO performance results of Regional Ocean Modeling System(ROMS) through NetCDF4[1] in parallel IO. The results show that the performance of parallel NetCDF4 is compatible with parallel NetCDF for atmospheric and oceanographic modeling system such as ROMS. It provides a certain degree of confidence for such modeling systems to use NetCDF4 in their applications.

I. Introduction

This report presents a performance study of Parallel NetCDF4[1], a collaborative work between University Corporation for Atmospheric Researcher (UCAR) Unidata Program and National Center for Supercomputing Applications (NCSA) HDF group. We study the performance of parallel NetCDF4 by measuring the IO performance of parallel NetCDF4 through Regional Ocean Modeling System(ROMS). This report also aims to assess parallel NetCDF4 performance and usability for MPI applications that use NetCDF as their IO package.

II. Regional Ocean Modeling System (ROMS)

The Regional Ocean Modeling System (ROMS) is a free-surface, hydrostatic, primitive equation ocean model that uses stretched, terrain-following coordinates in the vertical and orthogonal curvilinear coordinates in the horizontal [2]. The ROMS model can be run in parallel using the MPI standard package [3].

All input and output for the model is via NetCDF3 [4]. This paper reports investigations using netCDF4 to implement the ROMS history file writer in parallel.

III. Why Parallel NetCDF4?

NetCDF4 is a new version of NetCDF that has an option to use HDF5[5] as its data storage layer with the existing NetCDF3 APIs. The architecture of NetCDF4 can be illustrated by Figure 1. It was implemented by UCAR Unidata and NCSA HDF group(now a non-profit corporation "The HDF Group").The beta release of NetCDF4 is scheduled in Summer, 2007. The advantages of using NetCDF4 are as follows:

- NetCDF users can use new features provided by HDF5 such as:
  - Extensible datasets
  - Data compression
  - Complex data types such as struct or array datatypes
  - Parallel IO through MPI-IO
- Users can still use almost all the existing NetCDF3 APIs. Very few new APIs are added to NetCDF4.
- UCAR Unidata, which designed, implemented and maintained the original NetCDF3, will distribute and maintain NetCDF4.
- HDF5 applications can also have access to files created by NetCDF4 applications.

The possible disadvantage of using NetCDF4 is:

- Users have to install HDF5 library in addition to NetCDF library.

Argonne National Lab and Northwestern Univ. implemented a parallel version of NetCDF based on NetCDF version 3 data model (Hereafter parallel NetCDF). Both parallel NetCDF and parallel NetCDF4 support collective IO, an optimized IO mode provided by MPI-IO. Parallel NetCDF also provides hints to optimize the performance on some computing systems. We have evaluated the performance of parallel NetCDF in ROMS in an earlier report [6] and found that parallel NetCDF can significantly improve the IO performance. However, it also requires users to change their APIs and is unable to support multi-unlimited dimensions. Therefore, the parallel netCDF4 seems a reasonable alternative for some applications.

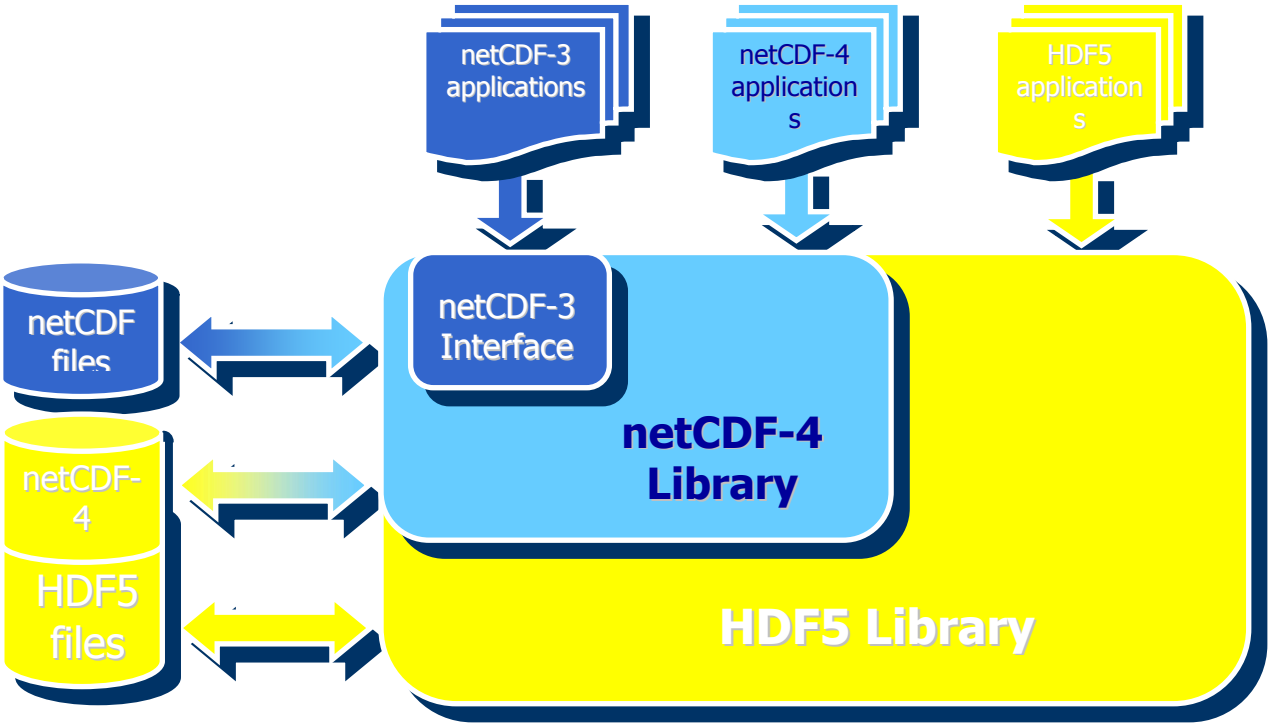


Figure 1: The architecture of NetCDF4

In this report we will implement a ROMS-parallel NetCDF4 history file writer with the same approach and settings as the ROMS-parallel NetCDF history file writer. We then compare the performance of parallel NetCDF4 with that of parallel NetCDF. We hope that the results can build up some confidences for potential parallel NetCDF4 applications.

IV. A Parallel NetCDF4-ROMS History File Writer

This study used ROMS version 2.0. The model has an option for us to use MPI for computation and uses sequential IO with Unidata NetCDF 3.5. We added another option to output the ROMS history file with parallel NetCDF4. We are using Parallel NetCDF4 alpha release for the study. During the time of report, we found that in the current version a default fill value will always be set by NetCDF4 even when no fill value is set by users. This causes the raw data IO time to be doubled. Unidata developers report this will be fixed in the NetCDF4 beta release. We explicitly turn off the setting of default fill value inside the NetCDF4 alpha library in this performance study.

Note that the history file is only one of many files used by ROMS. A full implementation would need to make additional changes to ROMS to read the history file and to support parallel IO for other input and output files.

V. Performance Report

This paper reports three limited performance experiments using ROMS with parallel NetCDF4. The system we are using is NCAR Bluesky, an IBM Cluster 1600 with AIX 5.1 and General Parallel File System(GPFS). Each node has 32 Power4 processors and 4 ports to the switch which provides 1 GB/s of bandwidth. Since we had to share computer resources with other users, we found that the data output time changed significantly under different model runs. For each experiment, three runs were done and the best (lowest) performance was chosen. The typical ROMS raw data are arrays with one to four dimensions. One of those dimensions is unlimited.

Table 1.

Characteristics of ROMS Output with the total size ~ 995 MB

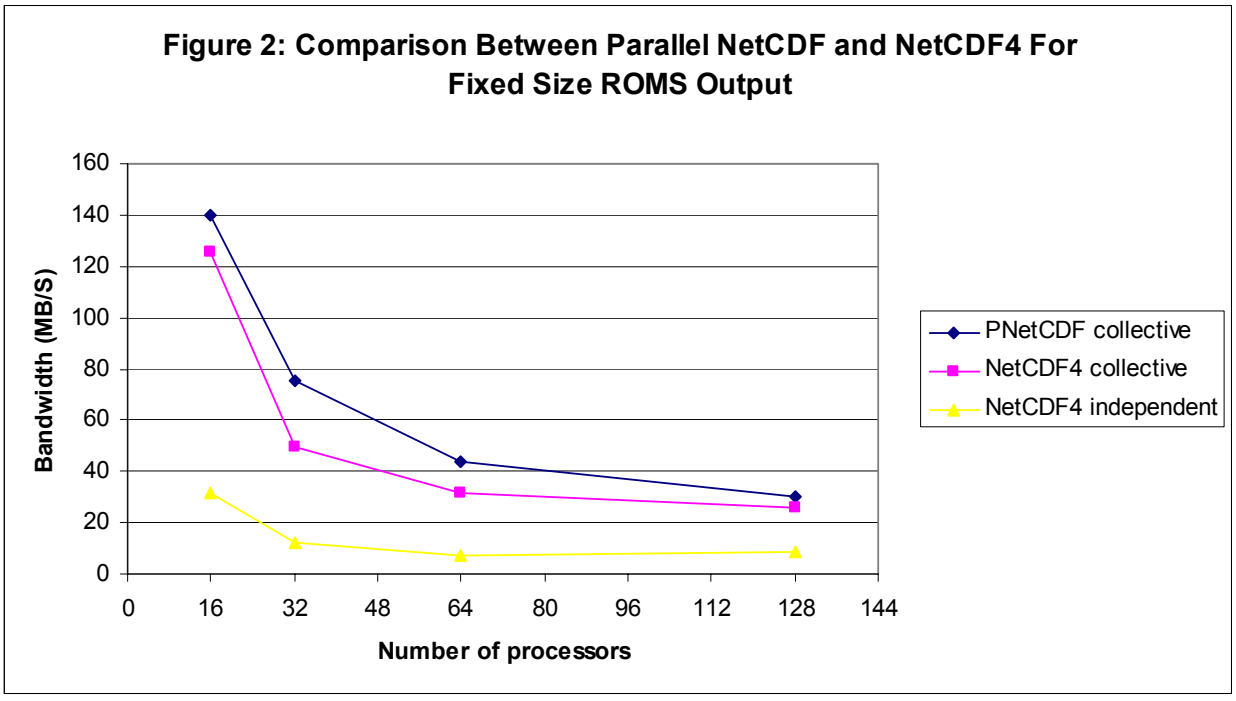
The maximum variable size ~ 150 MB

Variable	Number of variable
Size < =8 bytes	~30
8 byte < Size <= 10 KB	~20
10 KB < Size < =10MB	~10
>10 MB	~5

Experiment 1:
Performance Comparison Between Parallel NetCDF and Parallel NetCDF4 for ROMS Output in Fixed Size

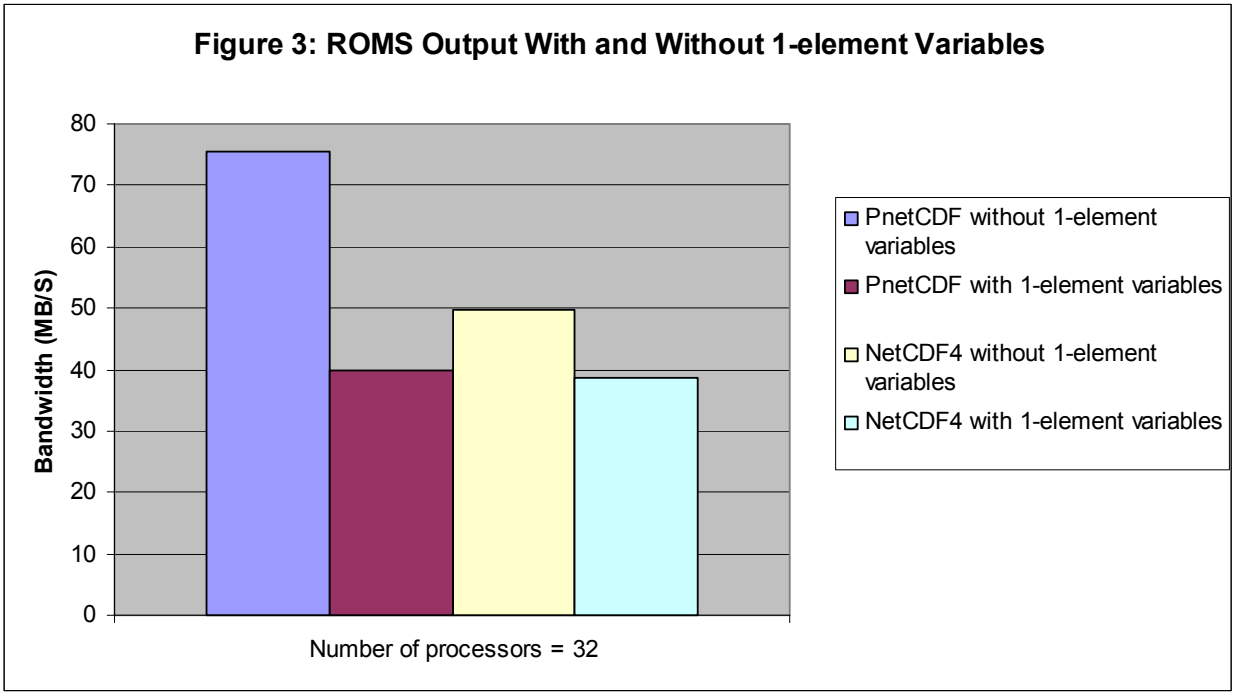
We have compared the performance among parallel NetCDF collective mode, parallel NetCDF4 collective mode and parallel NetCDF4 independent mode. The data characteristics can be seen in table 1. Figure 2 shows the comparison results of the aggregate bandwidth of ROMS parallel IO in both libraries. The performance of parallel NetCDF4 in collective mode is close to the performance of parallel NetCDF in collective mode with an average 15% performance degradation. This is acceptable with the consideration of the additional overhead caused by adding the NetCDF4 layer on top of HDF5. The performance with independent IO in parallel NetCDF4 is much worse than those in collective IO. This is consistent with our previous performance study[7].

One seemingly unexpected result is that the aggregate bandwidth decreases as the number of processors increases. We believe that this is partly due to the characteristics of ROMS output. Many small IO writes causes greater communication overhead as the number of processors increase. Another reason is due to the hardware settings. The significant decrease in performance present during the transition from 16 to 32 processors may be caused by a larger demand of resources while the number of switch ports remains constant (32-way node). This is consistent with the previous parallel NetCDF performance report[6] and the pattern is similar for both parallel NetCDF and parallel NetCDF4. This suggests that using more processors does not always generate better IO performance.



Experiment 2:
Performance Comparison Between Parallel NetCDF and Parallel NetCDF4 with and without small IO writes

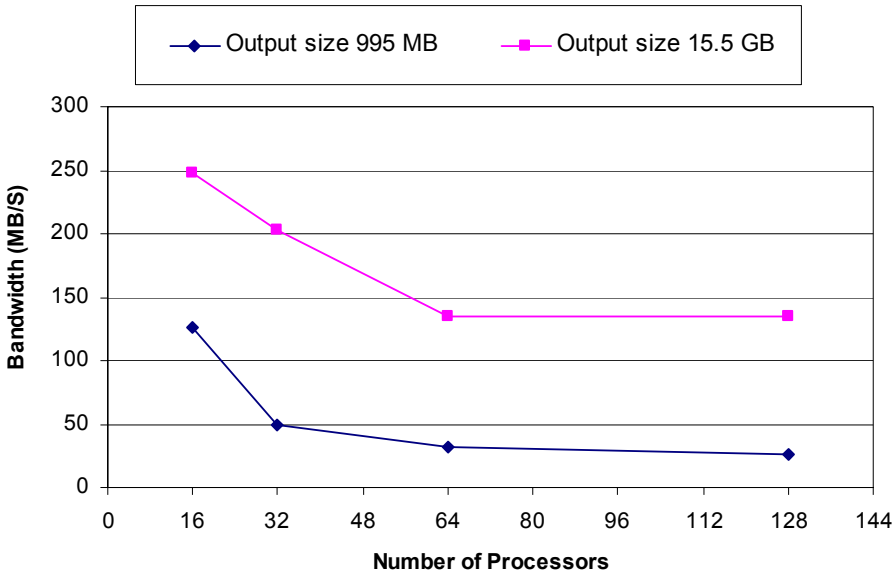
This experiment shows how small writes affect the ROMS IO performance for parallel NetCDF and parallel NetCDF4. ROMS is run with and without outputting the 32 1-element data using 32 processors. As shown in Figure 3, performance degrades significantly for both parallel NetCDF and parallel NetCDF4 once running the mode with outputting the 32 1-element data. NetCDF4 can improve this case significantly by grouping these 32 1-element data into one 1-element of compound datatype that includes the 32 fields. This should significantly improve the IO performance. The big difference between parallel NetCDF and parallel NetCDF4 without 1-element variable seems to be caused by the sharing of computing system we are using to do performance.



Experiment 3:
Parallel NetCDF4 Performance for ROMS Output in Different Size

As discussed in the previous section, small IO writes may cause bad IO performance. This experiment tries to verify that parallel NetCDF4 ROMS history file writer can indeed improve the IO performance as the output file increases. Figure 4 shows that the IO performance can improve up to 300% as the output file size increases from 995 MB to 15.5 GB.

Figure 4: ROMS Output with NetCDF4 Collective



VI. Summary

1. We find that the performance of parallel NetCDF4 is compatible with parallel NetCDF with about 15% slowndown on average. We suspect that the slowness is due to the software management when passing information from parallel NetCDF4 to HDF5.
2. We also find that the IO performance using collective IO mode is much better than that using independent IO mode; with the speed-up factor of at least 2.
3. We find that with the increasing of the number of processes, the aggregated bandwidth actually decreases. We believe that this is partly because of many small IO writes (less than 10 KB) in ROMS and hardware settings in our experiments. We find that the overall performance improves as the output file size increases. This is consistent with the previous performance report.
4. We find that the appearance of about 32 1-element variables causes a great performance penalty, especially for parallel NetCDF. We strongly suggest that parallel IO application avoids generating such variables. If it cannot be avoided, NetCDF4 provides a way to create a struct to combine all those variables into one variable. A decent performance can be expected then.

VII. Some suggestions for future work

This work is a partial implementation. There is much more that must be done to provide full parallel IO for ROMS. This work includes:

1. Add parallel NetCDF4 IO writers for other types of ROMS output files
2. Implement parallel NetCDF4 IO readers for all types of ROMS input files
3. Test on other platforms, including larger numbers of processors and other architectures
4. Optimize one-element array reading and writing, e.g., by consolidating many small operations

Acknowledgements

The netCDF-4/HDF5 development project is supported by the NASA Earth Science Technology Office under NASA award AIST-02-0071.

References:

[1] Unidata, netCDF4: Merging the NetCDF and HDF5 Libraries, http://my.unidata.ucar.edu/content/software/netcdf/netcdf-4/index.html
[2] Regional Ocean Model System, Web site: http://marine.rutgers.edu/po/index.php?model=roms
[3] William Gropp, Ewing Lusk, Rajeev Thakur, 1999: Using MPI; The MIT Press.
[4] Unidata, netCDF, http://my.unidata.ucar.edu/content/software/netcdf/index.html
[5] The HDF Group, HDF5: http://hdf.ncsa.uiuc.edu/HDF5/
[6] Muqun Yang, Mike Folk, Robert E. McGrath, "Performance Study of Parallel NetCDF in ROMS", NCSA HDF group, August 27th, 2004, http://hdf.ncsa.uiuc.edu/apps/WRF-ROMS/Parallel_NetCDF_Performance.pdf
[7] Christian M. Chilan, MuQun Yang, Albert Cheng, Leon Arber, "Parallel I/O performance study with HDF5, a scientific data package, http://hdfgroup.com/HDF5/papers/papers/ParallelIO/ParallelPerformance.pdf