I/O Performance Isolation on A Shared Storage System for MPI-IO Applications

Yusuke Tanimura [†], Rosa Filgueira [††], Malcolm Atkinson [††] and Isao Kojima [†]

1. Introduction

The importance of data-intensive computing has been widely recognized in both science and industry. The traditional High-Performance Computing (HPC) systems are also expected to have further support for large scale data analysis. In such systems, MPI still dominates as a means of parallel programming and MPI-IO will take an important role to facilitate input/output of large data[1),2)].

MPI-IO provides an I/O interface for MPI applications, and coordinates I/O requests to parallel storage systems, such as PVFS2[3)], Lustre[4)] and so on, with optimizations such as data sieving[5)] and Two-Phase I/O[6)]. However, there is a problem in concurrent use of shared storage systems in large HPC systems[7)]. When more than two workloads simultaneously use the same I/O servers, storage devices, and/or storage network paths, their I/O performances interfere detrimentally. This might spoil performance improvement that might otherwise be obtained by optimizations of MPI-IO. In particular for parallel applications, one slow access causes a delay of the entire execution.

In order to take advantage of the optimizations of MPI-IO, this work presents I/O performance isolation between applications which use the same shared storage system, based on an advanced reservation mechanism. We are using Dynamic-CoMPI[8)] as a MPI-IO implementation and Papio[9)] as a shared storage system which implements parallel I/O and performance reservation. We are developing the ADIO layer to connect these systems and to evaluate the benefits of the reservation-based performance isolation approach.

† National Institute of Advanced Industrial Science and Technology (AIST)
†† The School of Informatics of the University of Edinburgh

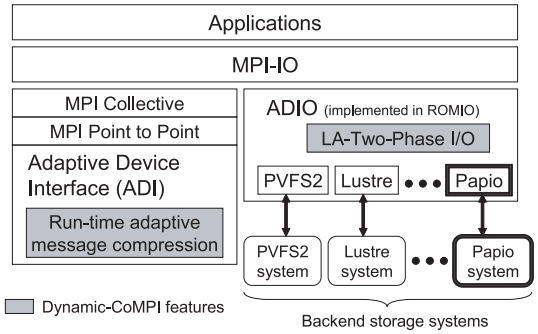

Fig. 1 Systems overview

2. Design

In our design, Dynamic-CoMPI and Papio are used for developing a proof-of-concept system, which examines how performance reservation works with MPI-IO applications.

Dynamic-CoMPI implements two optimization techniques in order to reduce the impact of communications and non-contiguous I/O requests in parallel applications. One is the locality aware strategy for Two-Phase I/O (LA-Two-Phase I/O) which optimizes data aggregation into contiguous buffers, and the other is run-time adaptive message compression. The benefit of LA-Two-Phase I/O was examined with PVFS2 as an underlying storage system.

The Papio storage system can allocate storage resources efficiently and control I/O processing priorities according to reservations. A reservation is made by application users, and I/O throughput (e.g. MB/sec), access type (write/read), start and end time of the access can be requested. The performance control is designed for large sequential I/O and therefore, it is potentially useful for many MPI-IO applications.

Dynamic-CoMPI is currently implemented into MPICH2[10)] which includes ROMIO. The ADIO layer of ROMIO is available for supporting other underlying storage systems. As shown

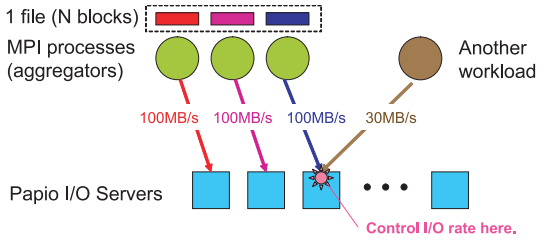

Fig. 2 N-to-1 layout of ad_papio

in **Fig. 1**, we are developing ADIO on Papio (ad_papio) to interface the two systems.

In ad_papio, the N-to-1 nonstrided data layout is used for collective calls of MPI-IO (e.g. MPI_File_write_all()). Thus one MPI process has one reserved access, and its reserved performance is the same for all of the processes. When another workload is assigned to the same storage server, the reserved performance is guaranteed as shown in **Fig. 2**. When the integrated accesses requires higher throughput than the rate one storage server can provide, the striping technique can be used internally in the Papio storage system.

The only thing application developers need to do to use ad_papio is to set a reservation ticket given by the Papio storage system and a few of Papio's parameters. The ticket file path and other parameters should be given by calling MPI_Info_set().

3. Status

An initial prototype of ad_papio has been implemented. We are first evaluating the basic performance of ad_papio using synthetic benchmarks and the BISP3D application. The BISP3D is a 3-dimensional simulator of BJT and HBT bipolar devices. The datatypes used in communications are the floating-point based MPI datatype. The application uses Two_Phase I/O to write the results into the underlying storage system. Second, we are evaluating our approach by comparing the performance of PVFS2, and Papio with reservation. The preliminary results show that Dynamic-CoMPI with Papio can finish the execution of the BISP3D application within an expected time even when another workload competes for I/O, without loosing the benefit of LA-Two-Phase I/O, **Fig. 3**. We will present the detail of the experiments in the poster presentation.

One of our future works is to integrate the I/O performance reservation mechanism with the resource scheduling of the modern HPC sys-

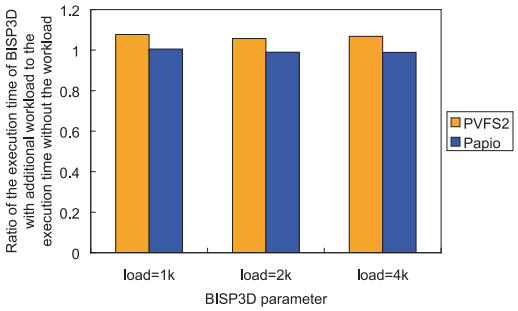

Fig. 3 Comparison results between PVFS2 and Papio with reservation

tems. We would like to free users from the additional operation for the reservation, and minimize a reserved time slot in each reservation to prevent overprovisioning.

Acknowledgments A part of this work was supported by KAKENHI (23680004).

References

1) Message Passing Interface Forum: MPI-2: Extensions to the Message-Passing Interface, http://www.mpi-forum.org/docs/docs.html.
2) Thakur, R. and et al.: Users Guide for ROMIO: A High-Performance, Portable MPI-IO Implementation, Technical ReportANL/MCS-TM-234 (1998).
3) PVFS: http://www.pvfs.org/.
4) Lustre: http://www.lustre.org/.
5) Thakur, R., Gropp, W. and Lusk, E.: Data Sieving and Collective I/O in ROMIO, *Proceedings of the 7th symposium on the Frontiers of Massively Parallel Computation (FRONTIERS'99)*, pp. 182–189 (1999).
6) del Rosario, J.M. and et al.: Improved Parallel I/O via a Two-phase Run-time Access Strategy, *ACM SIGARCH Computer Architecture News - Special issue on input/output in parallel computer systems*, Vol. 21, No. 5, pp. 31–38 (1993).
7) Lofstead, J. and et al.: Managing Variability in the IO Performance of Petascale Storage Systems, *Proceedings of SC'10* (2010).
8) Filgueira, R. and et al.: Dynamic-CoMPI: dynamic optimization techniques for MPI parallel applications, *The Journal of Supercomputing*, Vol. 59, No. 1, pp. 361–391 (2010).
9) Tanimura, Y. and et al.: A Distributed Storage System Allowing Application Users to Reserve I/O Performance in Advance for Achieving SLA, *Proceedings of Grid 2010*, pp.193–200 (2010).
10) MPICH2: http://www.mcs.anl.gov/research/projects/mpich2/.