An I/O Load Balancing Framework for Large-scale Applications (BPIO 2.0)

Sarah Neuwirth*, Feiyi Wang§, Sarp Oral§, and Ulrich Bruening*
*University of Heidelberg, Germany, {sarah.neuwirth,ulrich.bruening}@ziti.uni-heidelberg.de, §Oak Ridge National Laboratory, USA, {fwang2,oralhs}@ornl.gov

Problem Statement

- Large-scale scientific applications' usage patterns lead to I/O resource contention and load imbalance
- Implementation of a dynamic, shared library based on BPIO, a method to resolve contention, provides a transparent way to balance resource usage without source code modification or recompilation

Introduction

Balanced Placement I/O (BPIO) Library

- Topology-aware and balanced data placement [1]
- Resolves application-level I/O contention
- Computes placement cost for each I/O client based on a tunable, weighted cost function:

$$Placement\ Cost = w_1 * R_1 + w_2 * R_2 + \ldots + w_n * R_n$$

$$R_i: resource\ component \quad w_i: weight\ factor$$

- Available as a user space library, but requires direct integration into the source code

Aequilibro – Integrating BPIO with ADIOS

- ADIOS [2] provides portable, fast, scalable, easy-to-use, metadata rich output and I/O interfaces can be changed during runtime
- Aequilibro [3], [4] combines the optimization done at the interconnect level by BPIO with the benefits of the ADIOS I/O framework

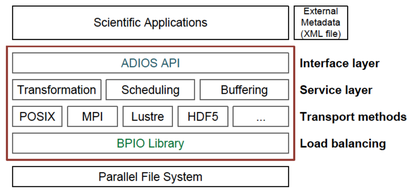

Fig. 1: Aequilibro software stack.

Research Objectives

- Design and implement a pre-loadable, shared library based on the BPIO library
- Evaluate the performance of the framework with a synthetic benchmark (IOR) for POSIX I/O and MPI-IO
- Evaluate with real-world HPC workloads on Titan

Aequilibro Performance Results

- IOR synthetic benchmark (POSIX and MPI-IO):
  - Measures parallel file system I/O performance
  - Setups: (I) Default, (II) BPIO, (III) ADIOS, (IV) Aequilibro
  - 10 scaled runs per test case on Titan, 3 repetitions per run
- Real-world HPC workload based on S3D physics code
- Metrics of interest:
  - Performance improvement with BPIO
  - Average bandwidth per second (GB/s)

$$Performance\ Improvement = 100 * \left(\frac{Bandwidth_{BPIO}}{Bandwidth_{Default}} - 1\right)$$ [1]

- Test System:
  - Titan Cray XK7 with 18,688 nodes
  - Spider II file system based on Lustre
  - 2x 144 Object Storage Servers (OSSs)
  - 2x 1,008 Object Storage Targets (OSTs)

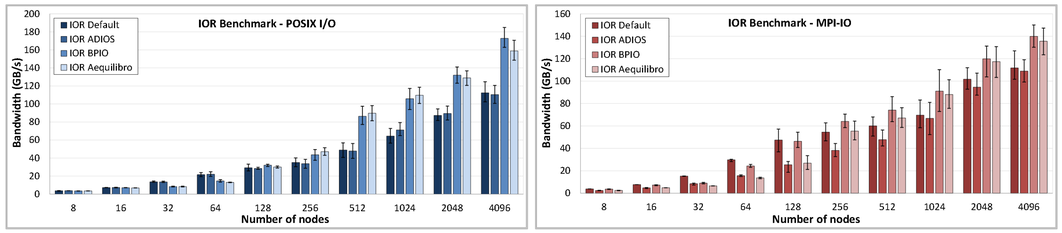

Fig. 2: IOR bandwidth performance for setup (I) to (IV) for POSIX I/O and MPI-IO including errors bars.

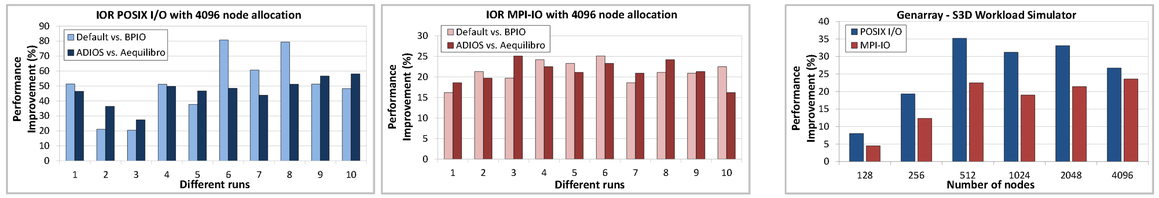

Fig. 3: Performance improvement for IOR (I) vs. (II) and (III) vs. (IV) at large-scale.

Fig. 4: S3D performance improvement.

Balanced Placement I/O Framework – BPIO 2.0

BPIO Runtime Environment

- Build as a shared, pre-loadable library (LD_PRELOAD)
- Utilizes BPIO library for balanced data placement
- Uses function interposition to prioritize itself over standard function calls
- End-to-end and per job load balancing
- Supported I/O interfaces include POSIX I/O and MPI-IO; HDF5 is under development

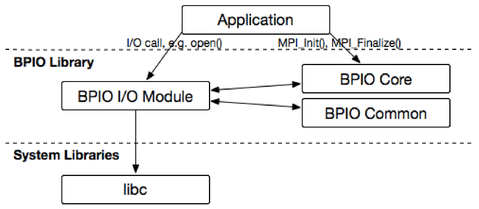

Fig. 5: BPIO runtime environment.

Dynamic Instrumentation

- Wrapper functions to intercept I/O functions
- Internal functions to initialize and maintain internal data structures and module-specific I/O characterization data
- Set of functions to interact with the BPIO library

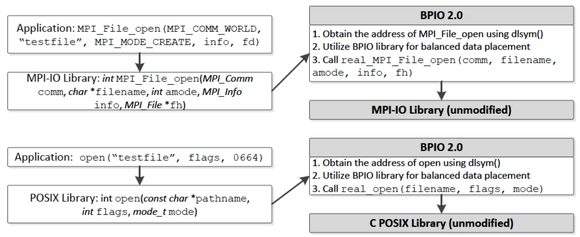

Fig. 6: Dynamic interception of I/O functions at runtime.

Experimental Results with BPIO 2.0

- Initial performance evaluation of the pre-loadable BPIO framework with the IOR benchmark
- Performance improvement and average bandwidth per second are similar to the direct integration of BPIO into an application

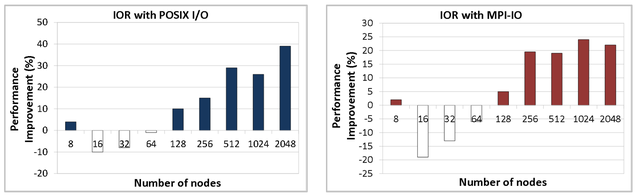

Fig. 7: Performance improvement for IOR BPIO 2.0 vs. IOR Default.

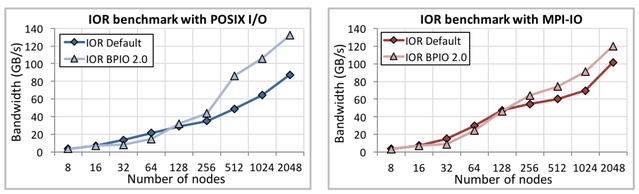

Fig. 8: Bandwidth performance for IOR Default and IOR BPIO 2.0.

Conclusions

- Aequilibro provides performance improvement, but requires the explicit BPIO integration into ADIOS' transport methods
- BPIO 2.0 is built as a dynamic library so it does not require any code modification or recompilation
- Initial experiments show similar performance improvement trends as a direct BPIO integration
- Ongoing and future work:
  - Single shared file support for MPI-IO
  - Extensive real-world HPC workload evaluation
  - Performance comparison of Aequilibro and BPIO 2.0
  - Support of HDF5

References

[1] F. Wang et al., *Improving Large-scale Storage System Performance via Topology -aware and Balanced Data Placement*, In ICPADS '14 (pp. 656-663).

[2] J. Lofstead et al., *Flexible IO and Integration for Scientific Codes through the Adaptable IO System (ADIOS)*, In CLADE '08 (pp.15-24).

[3] S. Neuwirth, S. Oral, F. Wang, Q. Liu, and S. Vazhkudai, *Improving Large-scale Application Performance with ADIOS and BPIO*, Poster at SMC '15.

[4] S. Neuwirth et al., *Using Balanced Data Placement to Address I/O Contention in Production Environments*, In SBAC-PAD '16.

U.S. DEPARTMENT OF ENERGY | Office of Science
This research used resources of the Oak Ridge Leadership Computing Facility, located in the National Center for Computational Sciences at the Oak Ridge National Laboratory, which is supported by the Office of Science of the Department of Energy under Contract DE-AC05-00OR22725.
OAK RIDGE INSTITUTE FOR SCIENCE AND EDUCATION Managed by ORAU for DOE
ORAU Further Together