A Framework for Auto-Tuning HDF5 Applications

Babak Behzad
University of Illinois at Urbana-Champaign

Joseph Huchette
Rice University

Huong Vu Thanh Luu
University of Illinois at Urbana-Champaign

Ruth Aydt
The HDF Group

Surendra Byna
Lawrence Berkeley National Laboratory

Yushu Yao
Lawrence Berkeley National Laboratory

Quincey Koziol
The HDF Group

Prabhat
Lawrence Berkeley National Laboratory

ABSTRACT

The modern parallel I/O stack consists of several software layers with complex interdependencies and performance characteristics. While each layer exposes tunable parameters, it is often unclear to users how different parameter settings interact with each other and affect overall I/O performance. As a result, users often resort to default system settings, which typically obtain poor I/O bandwidth. In this research, we develop a benchmark guided auto-tuning framework for tuning the HDF5, MPI-IO, and Lustre layers on production supercomputing facilities. Our framework consists of three main components. *H5Tuner* uses a control file to adjust I/O parameters without modifying or recompiling the application. *H5PerfCapture* records performance metrics for HDF5 and MPI-IO. *H5Evolve* uses a genetic algorithm to explore the parameter space to determine well-performing configurations. We demonstrate I/O performance results for three HDF5 application-based benchmarks on a Sun HPC system. All the benchmarks running on 512 MPI processes perform 3X to 5.5X faster with the auto-tuned I/O parameters compared to a configuration with default system parameters.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

Parallel I/O, Auto-Tuning, Performance Optimization, Parallel file systems

Keywords

H5Tuner, H5Evolve, H5PerfCapture, HDF5 Auto-tuning

1. INTRODUCTION

Our goal in this research is developing a benchmark-driven auto-tuning framework for identifying appropriate HDF5, MPI-IO, and Lustre settings on a given platform. Figure 1 shows an overview of our I/O auto-tuning framework with H5Tuner, H5PerfCapture, and H5Evolve. *H5Tuner* provides transparent parameter injection into the parallel I/O calls. It is a shared library which can be preloaded before the HDF5 library, prioritizing it over the native HDF5 functions. *H5PerfCapture*, built on Darshan [1], gathers I/O performance statistics, such as I/O time and number of bytes read/written, and traces HDF5 calls. *H5Evolve* uses a genetic algorithm (GA) to sample the I/O parameter space in order to find high-performing I/O configurations.

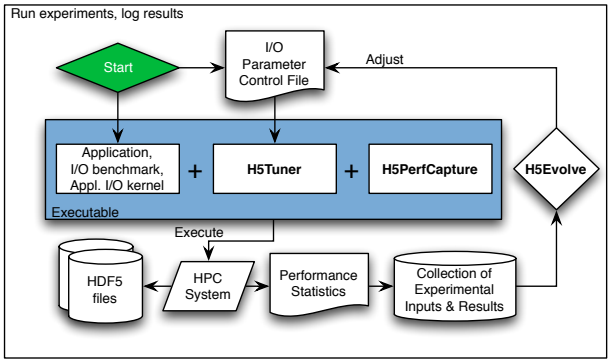

Figure 1: A functional schematic of the the auto-tuning framework

2. RESULTS

We choose three parallel I/O kernels to evaluate our auto-tuning framework: VPIC-IO, VORPAL-IO, and GCRM-IO. These kernels are derived from real scientific applications. We applied the auto-tuning framework for these applications on Texas Advanced Computing Center's Ranger system. We ran the tests on 128 and 512-core concurrency. We hand-selected a number of important parallel I/O parameters from the HDF5 (alignment), MPI-IO (collective buffer size, number of collective buffering nodes) and Lustre (strip count, stripe size) software layers.

Figure 2 shows the GA evolution of overall GCRM-IO kernel runtime using *H5Evolve* on 512 Ranger cores. The x-axis shows the experiment number and the y-axis shows the time taken to complete writing GCRM-IO data. We ob-

HPDC'13, June 17–21, 2013, New York, NY, USA.
ACM 978-1-4503-1910-2/13/06.

served a large variation in I/O time, with spikes corresponding to parameter choices that performed poorly. Over time, the GA adjusts tunable parameters to find good combinations, favoring exploration around well-performing choices. We chose the set of parameters with the smallest I/O time in the last group of experiments (the last GA generation) as the tuned set.

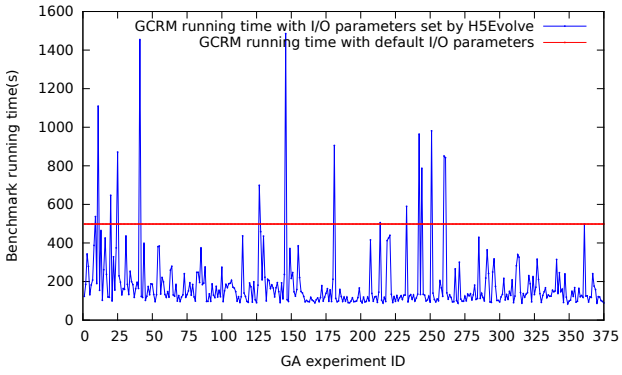

Figure 2: Evolution of GCRM-IO runtime with H5Evolve on Ranger using 512 cores

Tuned Parameters, Runtimes, and Speedup of Tuned over Default						
System	Ranger (128 Cores)			Ranger (512 Cores)		
Application	VPIC-IO	GCRM-IO	VORPAL-IO	VPIC-IO	GCRM-IO	VORPAL-IO
Parameter	Tuned Sets of Parameters Identified by H5Evolve					
Stripe Count	16	32	16	96	96	32
Stripe Size/ CB Buffer Size	16 MB	16 MB	32 MB	128 MB	1 MB	8 MB
CB Nodes	128	96	96	48	64	64
Alignment (thrsh, bndry)	0 KB, 4 KB	0 KB, 64 KB	0 KB, 16 KB	4 KB, 16 KB	0 KB, 64 KB	0 KB, 256 KB
Description	Measured Runtime (seconds) / Bandwidth (MB/s)					
Default Parameters	119.91	135.43	179.97	417.50	498.21	391.72
	258.52	302.73	322.27	308.98	327.17	584.60
Minimum	57.38	44.75	50.76	127.92	84.29	103.99
Maximum	243.88	284.26	357.54	1205.89	1485.36	959.51
Tuned Set	68.11	48.86	53.31	132.64	89.64	108.52
	455.14	839.13	1087.97	972.55	1818.38	2110.21
Speedup	**1.76x**	**2.77x**	**3.37x**	**3.14x**	**5.55x**	**3.60x**

Table 1: Tuned results for Ranger using 128 cores and 512 cores

Table 1 summarizes tuned I/O parameters, runtime, and speedup obtained by the framework for the kernels and platforms for three benchmarks using 128 and 512 cores. We can observe speedups ranging from 1.7x to 3.4x for 128-core scale and those ranging from 3.1x to 5.5x at 512-core scale compared to default I/O settings.

3. RELATED WORK

Auto-tuning has been used extensively in computer science for improving performance of computational kernels [4, 3, 5]. Our study focuses on auto-tuning *I/O subsystem* for writing and reading data to a parallel file system in contrast to tuning a few computational kernels. Yu et al. [7] manually characterize, tune, and optimize parallel I/O performance on Lustre file system of Jaguar. Howison et al. [2] also perform manual tuning of various benchmarks that select parameters for HDF5, MPI-IO and Lustre parameters on Hopper. You et al. [6] proposed an auto-tuning framework based on queuing theory models for Lustre file system on Cray XT5 systems at ORNL. They search for file system stripe count, stripe size, I/O transfer size, and the number of I/O processes. Developing a mathematical model for different systems can be farther from the real system and may produce inaccurate performance results. In contrast, our framework searches for parameters on real system using search heuristics.

4. CONCLUSIONS

We have presented a general framework for optimizing I/O performance of HDF5 applications. The framework is able to search a configuration space consisting of HDF5, MPI-IO and Lustre parameters to determine good settings. The framework is then able to execute these settings without requiring any effort from the application developer. We have demonstrated the successful application of the framework for three HDF5 benchmarks derived from production simulation codes. We applied the framework on a Sun Constellation cluster, and demonstrate convincing performance improvements over system default settings. We believe that this approach holds much promise in terms of hiding the complexity of the I/O stack and providing performance portability.

5. ACKNOWLEDGMENTS

This work is supported by the Director, Office of Science, Office of Advanced Scientific Computing Research, of the U.S. Department of Energy under Contract No. AC02-05CH11231. This research used resources of the the Texas Advanced Computing Center. The authors would like to acknowledge John Shalf, Mohamad Chaarawi and Marc Snir for their support and guidance.

6. REFERENCES

[1] P. Carns et al. Understanding and improving computational science storage access through continuous characterization. In *27th IEEE Conference on Mass Storage Systems and Technologies*, 2011.

[2] M. Howison et al. Tuning HDF5 for Lustre File Systems. In *Proceedings of 2010 Workshop on Interfaces and Abstractions for Scientific Data Storage (IASDS10)*, 2010.

[3] R. Vuduc, J. Demmel, and K. Yelick. Oski: A library of automatically tuned sparse matrix kernels. In *Proceedings of SciDAC 2005, Journal of Physics: Conference Series*, 2005.

[4] R. C. Whaley, A. Petitet, and J. J. Dongarra. Automated empirical optimization of software and the ATLAS project. *Parallel Computing*, 27(1–2):3–35, 2001.

[5] S. Williams et al. Optimization of sparse matrix-vector multiplication on emerging multicore platforms. In *2007 ACM/IEEE conference on Supercomputing*, SC '07, pages 38:1–38:12, 2007.

[6] H. You, Q. Liu, Z. Li, and S. Moore. The design of an auto-tuning i/o framework on cray xt5 system.

[7] W. Yu et al. Performance characterization and optimization of parallel i/o on the cray xt. In *IPDPS 2008.*, pages 1 –11, april 2008.