Restructuring Data for Exascale Scientific Productivity

Jay Lofstead[*] , Hasan Abbasi[‡] , Mark Ainsworth[‖‡] , Jong Choi[‡] , Matthew Curry[*] ,
Scott Klasky[‡§**], Tashin Kurc[‡‡], Qing Liu[‡] , Carlos Maltzahn[†] , Manish Parashar[¶] ,
Norbert Podhorzski[‡] , Feiyi Wang[‡] , Matthew Wolf[**], C. S. Chang[††], Michael Churchill[††],
Stephane Ethier[††]
[*] Sandia National Laboratories, [†] University of California Santa Cruz,
[‡] Oak Ridge National Laboratory, [§] U. Tenn. Knoxville, [¶] Rutgers University,
[‖] Brown University, [**]Georgia Tech, [††]PPPL, [‡‡]Stony Brook

ABSTRACT

As the exascale computing age emerges, data related issues are becoming critical factors that determine how and where we do computing. Popular approaches used by traditional I/O solution and storage libraries become increasingly bottlenecked due to their assumptions about data movement, re-organization, and storage. While, new technologies, such as "burst buffers", can help address some of the short-term performance issues, it is essential that we reexamine the underlying storage and I/O infrastructure to effectively support requirements and challenges at exascale and beyond.

In this paper we present - different data needs different techniques - write overhead for the different refactoring/auditing approaches - derived quantities in viz after precision based refactoring and differences from full fidelity data sets. - linear auditing approaches for data refactoring and the effects on data sizes, io performance, and viz quality. Do every point, but split at 24 bits and 40 bits.

1. INTRODUCTION

Be sure to address the following:
1. Why should Chang or Chen care from the apps side.
2.why should ken or hank care from the viz side
3.why would Garth care from the storage side

The rest of the paper is organized as follows. A brief overview of related work is presented first in Section 2. Section **??** discusses the programmatic interface end users will see when interacting with the storage array. Section **??** briefly discusses the motivation and proposal for the IO forwarding layer. Section **??** describes the IO Dispatcher layer and the broad functionality it offers. Section **??** discusses how the DAOS layer functions. The VOSD layer is discussed in Section **??**. Next is an exploration of cross-cutting features like transactions and metadata management in Section **??**. A demonstration of the functionality is presented in Section 3. The paper is concluded in Section 4 with a summary of the broad issues covered in the paper.

PDSW15 November 16, 2015, Austin, TX, USA

ACM 978-1-4503-2505-9/13/11...$15.00.
http://dx.doi.org/xxxx.xxx/xxxxxxxxxx

2. RELATED WORK

Many projects over the last couple of decades have sought to address some challenging aspect of parallel file system design. The recent rise of "Big Data" applications with different characteristic IO patterns have somewhat complicated the picture. Extreme scale machines will be expected to handle both the traditional simulation-related workloads as well as applications more squarely in the Big Data arena. This will require some adjustments to the underlying system for good performance for both scenarios.

The major previous work is really limited to full file systems rather than the mountain of file system refinements made over the years. A selection of these other file systems and some features that make it relatively unique are described below.

Ceph [8] is a distributed object store and file system. It offers both a POSIX and object interface including features typically found in parallel file systems. Ceph's unique striping approach uses pseudo-random numbers with a known seed eliminating the need for the metadata service to track where each stripe in a parallel file is placed. PVFS [2] offers optimizations to reduce metadata server load, such as a single process opening a file and sharing the handle. It has been commercialized in recent years as OrangeFS. Lustre [1] has become the de facto standard on most major clusters offering scalable performance and fine-grained end-user and programmatic control over how data is placed in the storage system. GPFS [6] offers a hands-off approach for providing good performance for scaling parallel IO tasks and is used extensively by its owner, IBM. Panasas [4] seeks to offer a dynamically adaptive striping system that detects the need for additional stripes for performance and adjusts the file layout as necessary.

Other file systems, like GoogleFS [3] and HDFS [7], address distributed rather than parallel computing and cannot be compared directly. The primary difference between distributed and parallel file systems is the ability of the file system to store and retrieve data simultaneously from multiple clients, in parallel, and treat the resulting collection of pieces as a single object. Distributed file systems rely on a single client creating a file, but distributing the set of files across a wide array of storage devices. The other, popular distributed file system of note is NFS [5] that has been

used for decades for enterprise file systems. These other file systems are mainly of interest in the context of the ACG features of FFSIO and will be discussed more in Section **??**.

3. DEMONSTRATION

4. CONCLUSIONS

The Fast Forward Storage and IO Stack project has designed a good first pass at addressing the requirements for an extreme scale data storage mechanism. By preferring a high level user API like HDF5 rather than using the POSIX interface, more advanced functionality can be incorporated with less end-user impact. The introduction of the IOD layer with buffering will absorb the difference between the compute node IO demands and the available bandwidth in the storage array. With DAOS supporting translating the container and object model to the underlying storage options, different storage technologies can be deployed over time.

With the overall stack design a prototype implementation complete, refinements, such as fault detection and recovery, can be designed and tested. These and other activities for phase 2 will ultimately generate what is likely to be the next generation storage stack for extreme scale platforms.

5. ACKNOWLEDGEMENTS

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

6. REFERENCES

[1] P. J. Braam. The lustre storage architecture. Cluster File Systems Inc. Architecture, design, and manual for Lustre, Nov. 2002. http://www.lustre.org/docs/lustre.pdf.

[2] P. H. Carns, W. B. Ligon III, R. B. Ross, and R. Thakur. PVFS: A parallel file system for linux clusters. In *Proceedings of the 4th Annual Linux Showcase and Conference*, pages 317–327, Atlanta, GA, Oct. 2000. USENIX Association.

[3] S. Ghemawat, H. Gobioff, and S.-T. Leung. The Google file system. In *Proceedings of the NineteenthACM Symposium on Operating Systems Principles*, pages 96–108, Bolton Landing, NY, Oct. 2003. ACM Press.

[4] Object-based storage architecture: Defining a new generation of storage systems built on distributed, intelligent storage devices. Panasas Inc. white paper, version 1.0, Oct. 2003. http://www.panasas.com/docs/.

[5] B. Powlowski, C. Juszczak, P. Staubach, C. Smith, D. Lebel, and D. Hitz. NFS version 3: Design and implementations. pages 137–152, 1994.

[6] F. Schmuck and R. Haskin. GPFS: A shared-disk file system for large computing clusters. In *Proceedings of the USENIX FAST '02Conference on File and Storage Technologies*, pages 231–244, Monterey, CA, Jan. 2002. USENIX Association.

[7] K. Shvachko, H. Kuang, S. Radia, and R. Chansler. The hadoop distributed file system. In *Proceedings of the 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, MSST '10, pages 1–10, Washington, DC, USA, 2010. IEEE Computer Society.

[8] S. A. Weil, S. A. Brandt, E. L. Miller, D. D. E. Long, and C. Maltzahn. Ceph: A scalable, high-performance distributed file system. In *Proceedings of the 2006Symposium on Operating Systems Design and Implementation*, pages 307–320. University of California, Santa Cruz, 2006.