Storage-driven Scheduling to Leverage Reliable Exascale Storage

Bradley W. Settlemyer
Computer Science and Mathematics Division

David A. Dillow, H. Sarp Oral, Feiyi Wang
Computing and Computational Sciences Division

Oak Ridge National Laboratory

To ensure that adequate storage system performance is available at Exascale, job schedulers, runtime systems, and process schedulers need to be re-designed to become storage-driven. Within Exascale computing systems, the storage service will continue to be extremely reliable, and as the only reliable component within the computing infrastructure, scientific applications will rely on storage system checkpointing to ensure progress. Whether checkpoints are coordinated, application-directed snapshots, or collected as virtual machine images, the primary challenge is designing systems software to address I/O bottlenecks and satisfy the Exascale performance requirements.

Although emerging fast, solid-state media will improve storage system performance, the primary impediments to storage performance at scale are the competitive and interfering workloads that cause storage controllers to perform additional work, exemplified by mechanical disk seeks, write amplification, and garbage collection processes. The storage programming model requires a complete reconsideration to achieve Exascale performance levels. Existing storage APIs should be replaced with an expressive, declarative API that is asynchronous. To fully leverage the asynchrony, the storage system will service entire applications using a time-division multiple access (TDMA) scheme. This provisions dedicated access to a portion of the storage system for a period of time, providing quality of service guarantees to the application.

By combining asynchronous completion with storage system QoS guarantees, applications are able to avoid stalling computation while waiting for storage requests to complete. TDMA, a popular technique for ensuring consistent performance in cellular networks, enables the storage scheduling service to offer the application dedicated storage access at defined intervals, ensuring no concurrent application generates an interfering storage workload (e.g. an application performing small random writes to a storage server will dramatically lower the performance for an application that is attempting to perform a large sequential read simultaneously). In the storage-driven paradigm, the storage system interacts with the job scheduler to ensure compatible QoS profiles are co-scheduled, it notifies the runtime system of the temporal and topological boundaries of the dedicated storage access intervals, and it notifies the application of request completions and feedback about the utilization of QoS quanta.

Challenges Addressed. While scalable, memory-based checkpoints may be useful to achieve fault tolerance for certain classes of faults [moody10], Exascale applications must checkpoint and restore from persistent media to achieve robust fault tolerance in the face of partitioned networks, power failures, and environmental hazards. Inefficient storage access by applications leads to large variances in performance due to the lack of QoS guarantees.

Maturity. Oak Ridge National Laboratory's (ORNL) Spider File System is representative of the storage needs of current capability-computing systems, using 13,440 disks to provide 10PB of capacity. The total amount of stored information is estimated to be in excess of 240 Exabytes [hilbert11], and truly large storage systems, such as Amazon's Simple Storage Service (S3), currently host in excess of a trillion objects (and thought to be in excess of

500,000 disks). Despite experiencing three high profile outages in the last five years [malik12], S3 guarantees a 99.95% uptime, or 1.83 days of outage per year. Spider, with considerably less infrastructure than S3 (e.g. no geographic co-location to prevent outages due to catastrophic power loss), has provided 99.8% uptime – which compares well to modern capability-class computers. Current leading edge storage systems are already well equipped to handle Exascale reliability and capacity requirements. The remaining obstacle to achieving application reliability via Exascale storage systems is meeting the performance demands.

Uniqueness. Applications accessing existing leading edge storage systems achieve high levels of storage performance by leveraging the Map-Reduce programming model [dean08]. Web service providers, commercial analytics, and financial engineering applications have all embraced the "move code, not data" paradigm, an extremely scalable computational model that allows software developers to deploy embarrassingly parallel, data-intensive jobs with relatively ease. Unfortunately, the Map-Reduce semantic is not appropriate for scientific applications. Scientific codes routinely produce large amounts of data, such as checkpoints and visualization data, using highly coordinated, tightly coupled, computationally intensive kernels. The computational state to be stored is dependent upon the chain of previous states generated – a generational dependency not supported by Map-Reduce.

Novelty. The lightweight file system (LWFS) leveraged the idea of server-initiated I/O to improve storage system performance and scalability [oldfield06-1, oldfield06-2]. Efforts in storage QoS have focused on TDMA-based disk scheduling to ensure local file system clients receive guaranteed performance levels [brandt05]. Our approach extends both concepts simultaneously, envisioning a storage-driven programming model that interacts with the cluster, application, and storage device schedulers.

Applicability. Current data-intensive analysis jobs are prevented from leveraging the benefits of scalable numerical methods due to simplistic nature of Map-Reduce and the underlying Key-Value stores. By constructing a storage-driven scheduling scheme that provides QoS guarantees, the development of new, compute-intensive statistical data analysis routines may be applicable in commercial finance, computer-based medical diagnostics, and other data intensive fields that require sophisticated computational analysis.

Effort. As discussed earlier, existing storage architectures presently meet the reliability and capacity demands of Exascale systems. Building a decentralized storage service that provides dedicated, multiplexed access to the storage system based on feedback from applications, runtimes, and job schedulers is the primary effort for realizing this system. The popularity of HPC I/O middleware allows existing applications to use the new system without extensive modification. However, to derive full benefit from the storage-driven programming model, applications will require modification. We expect these modifications to only impose an incremental increase in the levels of effort beyond those required to accommodate asynchronous communication for Exascale software.

References

[brandt05] J. Wu and S. Brandt, "QoS Support in Object-based Storage Devices," International Workshop on Storage Network Architecture and Parallel I/O *(SNAPI 05)*, held in conjunction with the International Conference on Parallel Architectures and Compilation Techniques *(PACT 2005)*, Saint Louis, Missouri, September 17–21, 2005.

[dean08] J. Dean and S. Ghemawat. MapReduce: simplified data processing on large clusters. *Communications of the ACM* 51, 1 (January 2008), 107-113.

[hilbert11] M. Hilbert and P. Lopez, The World's Technological Capacity to Store, Communicate, and Compute Information, Science, 1 April 2011, Vol, 332 no. 6025 pp. 60-65.

[laney01] D. Laney 3-D Data Management: Controlling Data Volume, Velocity, and Variety, META Group Report, February 2001.

[malik12] O. Malik, Parts of Amazon Web Services Suffer an Outage, http://gigaom.com/cloud/did-amazons-web-services-go-down/

[moody10] A. Moody, G. Bronevetsky, K. Mohror, and B. de Supinski. Design, Modeling, and Evaluation of a Scalable Multi-level Checkpointing System. In *Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis* (SC '10). IEEE Computer Society, Washington, DC, USA, 1-11.

[oldfield06-1] R. Oldfield, P. Widener, et al., Efficient Data Movement for Lightweight I/O, Proceedings of the 2006 Workshop on High-performance I/O Techniques and Deployment of Very-Large Scale I/O Systems, Barcelona, Spain, 2006.

[oldfield06-2] R. Oldfield, A.B. Maccabe, et al., Lightweight I/O for Scientific Applications, Proceedings of the IEEE International Conference on Cluster Computing, Barcelona, Spain, 2006.