Paper 1815-2014

A Case Study: Performance Analysis and Optimization of SAS® Grid Computing Scaling on a Shared Storage

Suleyman Sair, Brett Lee, Ying M. Zhang, Intel Corporation

ABSTRACT

SAS® Grid Computing is a scale-out SAS® solution that enables SAS applications to better utilize I/O and compute intensive computing resources. This requires the use of high-performance shared storage (SS) that allows all servers to access the same file system. SS may be implemented via traditional NFS NAS or clustered file systems (CFS) like GPFS. This paper uses the Intel® Enterprise Edition for Lustre[1] (IEEL) file system, a parallel, distributed CFS, for a case study of performance scalability of SAS Grid Computing nodes on SS. The paper qualifies the performance of a standardized SAS workload running on IEEL at scale. Lustre is traditionally used for large block sequential I/O. We present the tuning changes necessary for the optimization of IEEL for SAS applications. In addition, results from the scaling of SAS Cluster jobs running on IEEL are presented.

INTRODUCTION

Around the world and across all industries, high-performance computing (HPC) is being used to solve today's most important and demanding problems. More than ever, storage solutions that deliver sustained high throughput are vital for powering HPC and "Big Data" workloads. As storage costs and challenges such as capacity and availability continue to grow, mining the value from the stored data becomes even more important. Further, today's high performance storage solutions are not COTS solutions but instead require custom and optimized solutions designed for the demanding needs of the customer. IEEL[1] is a high performance distributed file system that is ready to meet the challenges. From its roots in HPC, IEEL is now expanding to meet the needs of data storage in the government, public, and private sectors due to its maturity and capabilities. Backed by USG Fast Forward funding, IEEL is also leading the industry into the future of storage capacity - Exabyte storage.

The Lustre file system provides reliable data storage with unsurpassed performance and scalability. It is both open source and hardware-agnostic, making it suitable for working well with a variety of workloads on a broad set of hardware configurations. For example, when used with commodity-based hardware from any storage vendor and with Intel commercial support, Lustre allows users to readily access large datasets. The inherent performance characteristics of Lustre allow it to collect, and store data under one global namespace. As a result, the Lustre file system provides an affordable, high performance computing solution for data, in both LAN and WAN environments. Storage costs are further reduced with the Hierarchical Storage Management (HSM) capabilities as a new feature of Lustre.

IEEL has the performance, scalability and flexibility to meet customer needs at an affordable price point. To further complement customer's in-house support personnel, Intel offers IEEL with a commercial support option to speed up implementation and problem resolution via Intel's wide Lustre partner network. The improved manageability provided by Intel® Manager for Lustre and the Intel® Hadoop Adaptor for Lustre allows Intel Hadoop users seamless access to data on Lustre file systems.

In this paper, we evaluate the performance of the SAS Calibration Workload (SCW) on IEEL. Our evaluation platform consists of SAS 9.4 and IEEL 1.0 running on RHEL 6.4. We have 8 client nodes, each with 6 Intel Westmere EP processors, connected with a 20Gb/s network.

Our results show that IEEL is an effective storage medium for workloads running under SAS Grid. Initial runs of the SCW on the basic, un-tuned Lustre were not optimal. By default, tunable Lustre parameters are optimized for the typical HPC workload - large, sequential IO. After applying appropriate changes, the researchers were able to lower execution time of the SCW to a very competitive execution time. Results of these tests have been fed back to IEEL developers to provide additional enhancements that will further optimize the performance of the SCW on IEEL.

[1] Other names and brands may be claimed as the property of others.

INTEL ENTERPRISE EDITION FOR LUSTRE

Lustre (including IEEL) [2] runs on most of the large Linux* clusters in the world and is used by many HPC and commercial customers. While Lustre servers are currently only supported on Red Hat Enterprise Linux (and some derivatives), support of Lustre clients is extended to SLES 11 releases. The full support matrix of the Lustre version tested, version 2.4, is shown below.

Server Support	Client Support
RHEL 6.4	RHEL6.4
CentOS 6.4	CentOS 6.4
	SLES 11 SP2 (SUSE)
	FC/18 (Fedora)

Table 1. Lustre 2.4.0 Test Matrix

Lustre 2.4, the first release in the current maintenance stream, was released in May of 2013. The latest Lustre maintenance release is version 2.4.2, and the latest Lustre feature release is version 2.5.

Lustre uses object-based storage in which each file or directory can be thought of as an object with attributes. These objects reside on the underlying file system, currently either ext4 or ZFS. A file on a Lustre file system is spread across both object storage servers (OSS) and Metadata servers (MDS). A MDS has a single LUN that is called a Metadata Storage Target (MDT), while each OSS typically has multiple LUNs which are referred to as Object Storage Targets (OSTs). A Lustre file is typically comprised of multiple objects, one per OST, as well as the metadata (on the MDT) associated with that file. Files are assigned typical POSIX attributes such as file type and access permissions, but each file is also assigned (either explicitly or by default) Lustre attributes such as an OST striping pattern or stripe size. Lustre also supports storage pools within a file system, allowing finer control on where the objects of a file can reside. Altogether, this creates a separation of functionality between computing and storage for efficient use of resources and flexibility. To the end user application, Lustre appears as a locally mounted POSIX file tree.

Several factors allow Lustre to provide a high performance, expandable, shared- file global namespace with manageable storage. First, Lustre's separation of the data and metadata is critical in high capacity file systems. The file system efficiently handles how and where the data is stored and who has access. By processing a file's metadata separate from the data, the file system provides a direct pipe to large data files while processing other requests uninterrupted and this is the key to Lustre's almost linear scaling. Second, by allowing the file system to handle backend storage, the user does not need to know exactly where data resides; he only needs to access data in a familiar fashion. Lustre builds upon RAID architecture created to increase I/O performance by placing the I/O requests into a stripe of data chunks to "Just a Bunch of Disks" (JBOD). This provides a parallel I/O path to the disks for increased I/O bandwidth and performance over previous non-parallel file systems. Lustre builds on this by creating multiple OSTs from individual RAID'd LUNs. The Lustre file system stripes the data across the OSTs similar to a RAID-0 stripe across disks to make increases in storage bandwidth possible. The OSTs are grouped onto OSSs to provide sufficient back-end throughput to saturate the typically faster front-end network. By increasing storage capacity through the addition of OSS/OST units, the overall system benefits with increased performance at negligible cost to the management of the global namespace, since adding capacity adds bandwidth without significantly increasing management needs.

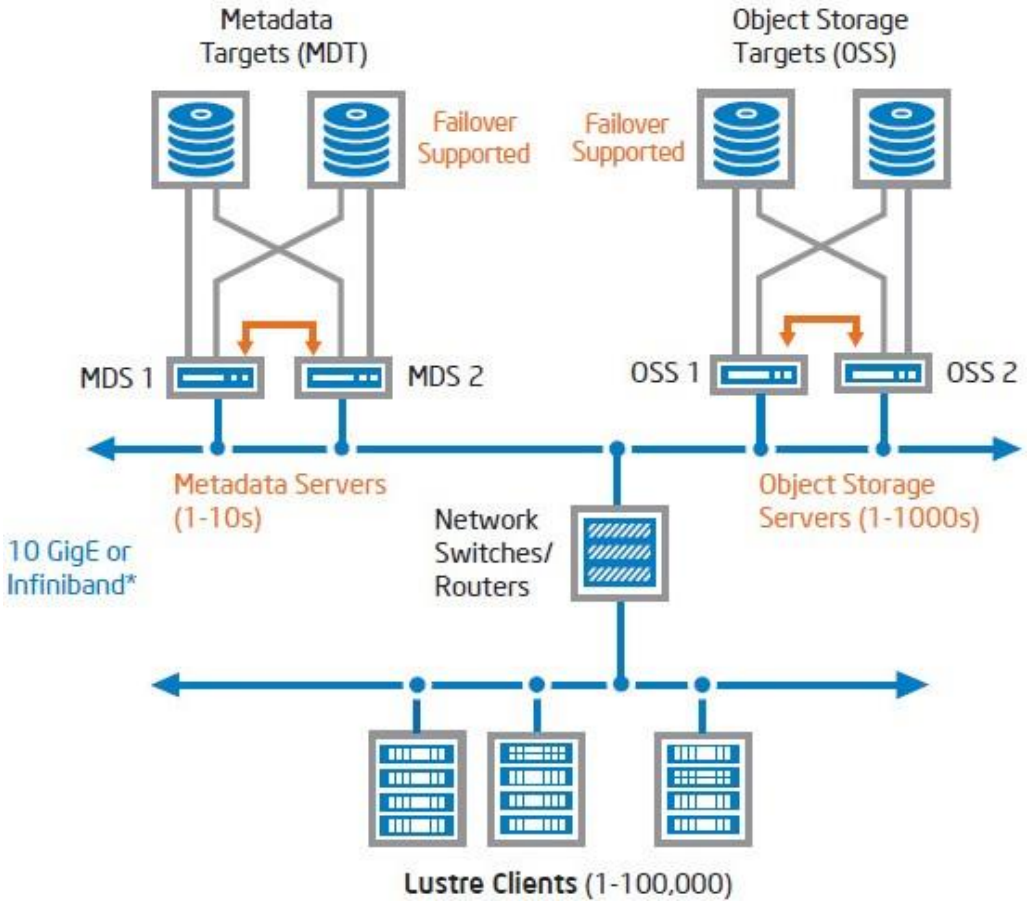

Figure 1. Lustre File System Configuration

A Lustre storage cluster consists of a Lustre Management Server (not shown), a Lustre MDS and several Lustre OSSs, each of which has associated disk storage as shown in **Figure 1** - Lustre File System Configuration. Client systems access these servers through the Lustre network (LNet). LNet is an efficient, RPC-based protocol that runs atop high performance networks; today those networks are typically Ethernet and InfiniBand. Lustre block operations bypass the MDS completely and utilize the parallel data paths to all OSTs in the cluster. Like other UNIX and Linux file systems, Lustre files are represented by inodes, but in the Lustre file system, these inodes contain references to the objects storing the file data. By default, Lustre stores file metadata in a 2KB inode, resulting in each file requiring an additional 2KB on top of the actual data stored in the file.

SAS/INTEL INTEGRATED PLATFORM

On a fully integrated SAS/Intel platform, the SAS software stack runs on top of IEEL and (potentially) Intel Distribution of Apache Hadoop, seamlessly to the SAS user. If utilizing Intel SSDs, the storage performance can further be optimized by including the Intel Cache Acceleration Software. This integrated platform is shown in Figure 2.

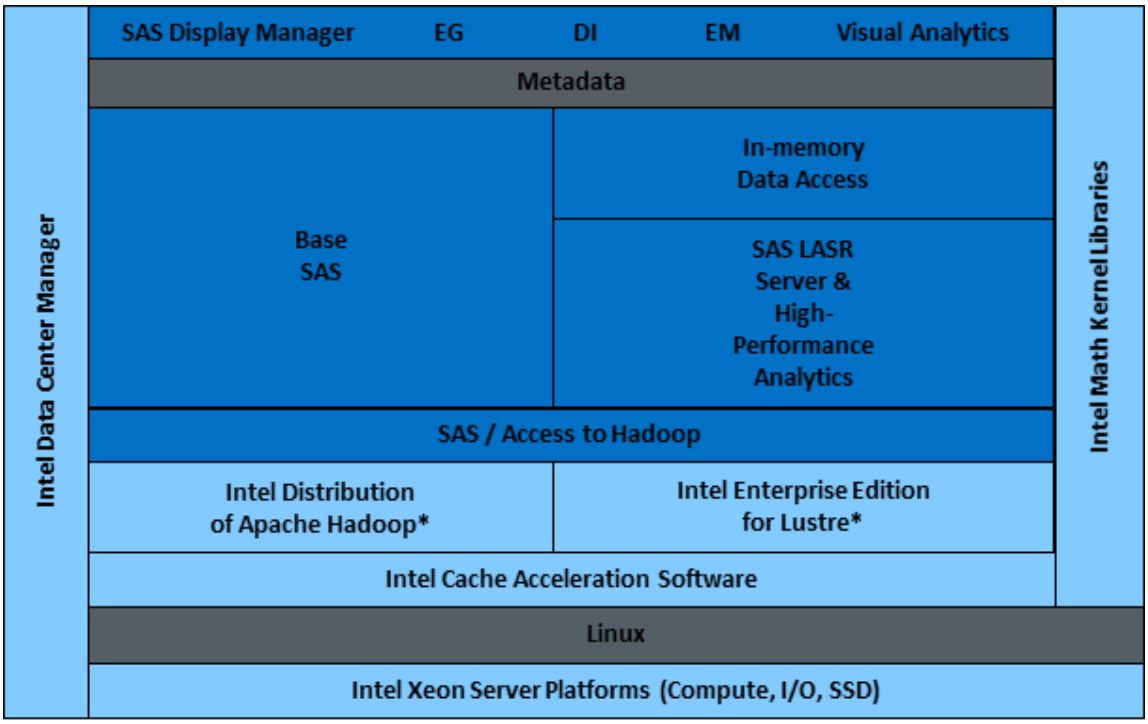

Figure 2. SAS/Intel Integrated Platform

SAS CALIBRATION WORKLOAD

The SAS Calibration workload is representative of a real SAS customer's workload and is used by SAS to evaluate storage subsystems for their analytics products. This workload runs a configurable number of concurrent SAS processes that execute a demand model calibration process. The workload emulates a SAS solution deployed in a distributed environment using shared input data and a shared output analytic data mart. The SAS programs are written using SAS macro language and create many small to moderate datasets.

There are 750 models in the workload and each model is calibrated by an individual SAS process. Each of these processes creates about 100 permanent output datasets and 7,000 temporary files. Many solutions, including SAS® Drug Development, SAS® Warranty Analysis, SAS® Enterprise Miner and others behave in a similar manner. This workload will exercise the management of file system metadata and some shared file systems have failed with what we consider a representative load. This workload tends to benefit a great deal from the file cache, because there are many small files that are read multiple times.

There are three types of files in play in this workload:

• **Read only fact tables:** There are a set of read only fact tables. These are written to the file system just after it is created and are not modified. Each process extracts a unique subset of data from the fact tables in a skip/sequential manner. Each subset is then used as input to the model calibration process.

• **Small temporary files:** Typically, accessing the fact tables consumes a large portion of the time to calibrate each model. After creating subsets of data from the fact tables, the analytic processes create and delete thousands of small files, typically under several MB each. This behavior is very typical of many SAS applications. There are many small files that are overwritten many times during the analytic processes. Also, it is now known what the size of a file will be, so the file is extended as we are writing it. Typically the page size 64k or 128k.

• **Analytic data mart:** This contains slightly larger files that are written once. Each of the 750 processes writes about 100 of these files each. These files are purged after the benchmark completes and are recreated the next time the benchmark runs.

EVALUATION PLATFORM

Our evaluation platform consists of several IEEL servers and 8 client nodes. Each IEEL server is a dual-socket 2.66GHz Westmere EP platform with 4-cores/8-threads and 12MB L2 cache on each socket. Each client node is a dual-socket 2.93GHz Westmere EP platform with 6-cores/12-threads and 12 MB L2 cache on each socket. Figure 3 depicts the cluster configuration diagram.

All the systems run RHEL 6.4, and SAS 9.4. In addition, IEEL 1.0 is installed on the systems. IEEL 1.0 consists of Lustre 2.3.11 servers and Lustre 2.4.0 clients. IEEL is configured with a single MDS and 3 OSS's. The MDT uses 4 SSDs in Raid10 configuration. The OST's are run on 15k rpm SAN array. The nodes are interconnected with a 20Gb/s Ethernet network. Table 2 shows the system configuration.

OS	**Red Hat* Enterprise Linux 6.4**
Application	SAS® 9.4
Workload	SAS® Calibration Workload
Platform	2 socket 2.66GHz Westmere EP platform for 4 server nodes 2 socket 2.93GHz Westemere EP platform for 8 client nodes
Memory	24 GB
Networking	Bonded (2x/3x) 10 Gb Ethernet
Shared Distributed File System	Intel Enterprise Edition Lustre 1.0
Drives in Storage Target	192 x 15K RPM Disks SAN array

Table 2. System Configuration

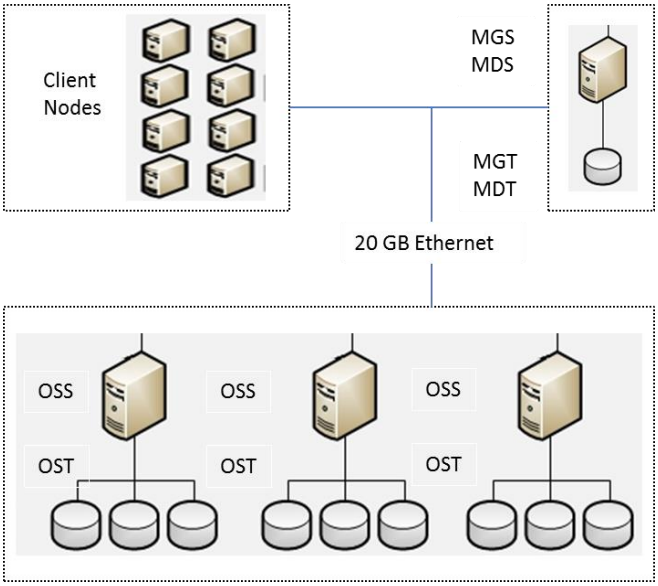

Figure 3. Diagram of Cluster Configuration

RESULTS

Measurement results are shown in Figure 4. The graph presents total execution time on the vertical axis and the number of concurrent jobs running on a single node on the horizontal axis. The color of each bar represents the number of IEEL mount points the experiment was run with. As mentioned earlier, having multiple mount points on each client emulates the DNE feature and alleviates the single MDT bottleneck.

When the 3 mount point experiment with 192 concurrent threads was repeated with a 30Gb/s network instead of the default 20Gb/s configuration, the calibration workload ran to completion in 77 minutes making this the best data point. This indicates that IEEL performance scales well with increased network bandwidth. In addition, when the number of client nodes is reduced to 5 from the original 8, the workload completes in 122 minutes, showing that IEEL scales well with the number of client nodes.

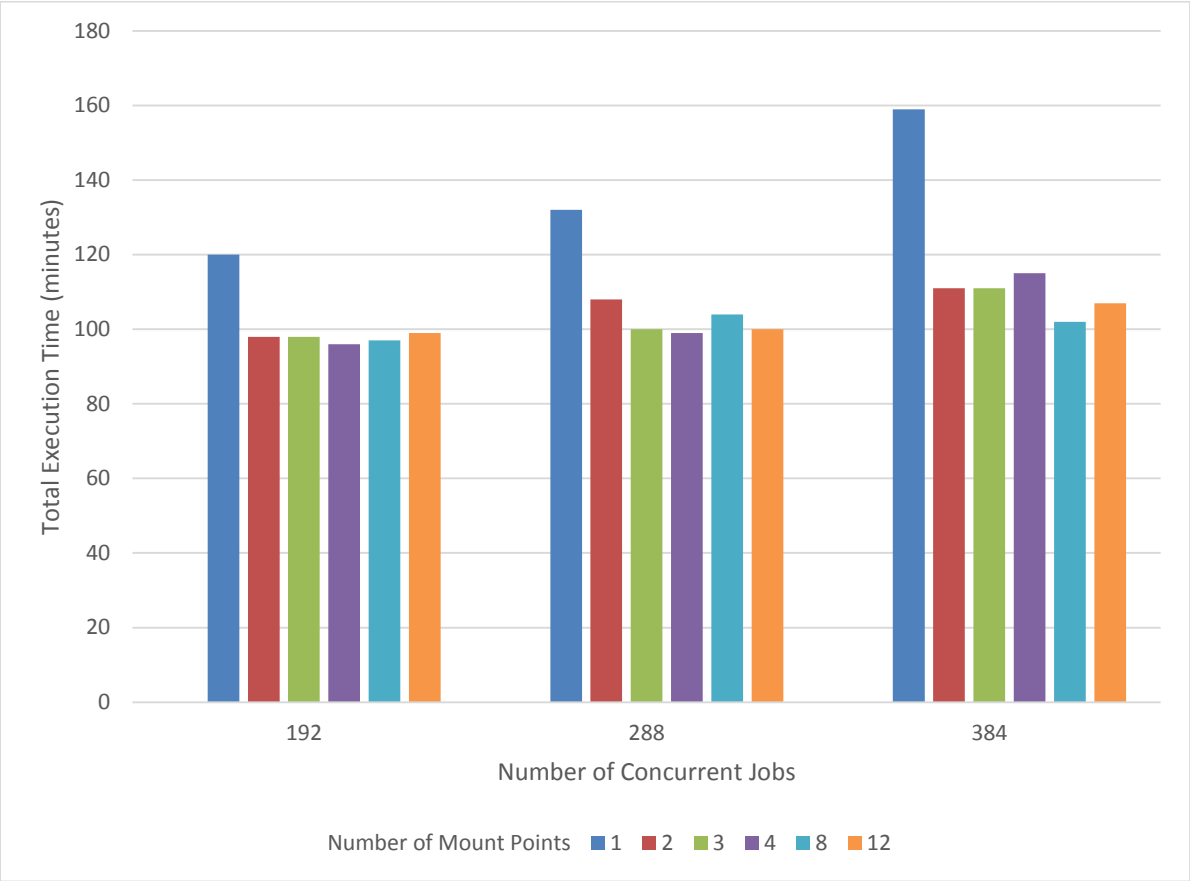

Figure 4. Calibration workload total execution time when varying the number of concurrent nodes.

DISCUSSION

When the SAS calibration workload was run using IEEL file system, the initial performance (wall time to completion) did not compare well to other parallel file systems. IEEL has a significant number of tunable parameters to increase performance, so several rounds of tests were executed with positive, but limited gains in performance. Given that SAS's calibration workload is more metadata and small file driven than the typical HPC workload, it was theorized that the immediate bottleneck to improving performance was the metadata path from client to metadata server (clients currently use a single-threaded RPC to communicate with the MDS). In an effort to test this theory, the IEEL file system was mounted multiple times on each client, with each new mount providing one additional metadata client, and thus one more RPC. Successive runs using incremental mount significantly improved performance, indicating that the metadata client was a significant performance bottleneck.

As Lustre was originally designed for sequential block IO, the IEEL Engineers at Intel have long recognized that both metadata and small file performance were areas that could be improved. To that end, the 2.4 release of Lustre added support for multiple (up to 4096) active MDT's. This new feature, Distributed NamespacE (DNE), spreads the metadata out across additional MDTs, with the goal of scaling out the metadata performance. A subsequent project, DNEv2, will bring additional metadata improvements beyond DNEv1. Regarding improving the performance of SAS with regard to small file IO, another new IEEL feature that is underway is often called "data on MDT." This project will have small (< 1MB) of block data on the MDT, so that when a small file is accessed it will only take one RPC to return both the metadata and the block data. Each of these new features, DNE and data on MDT, should significantly improve the performance of SAS on the IEEL file system. In combination with other tunable parameters [3] such as caching and readahead, the IEEL file system should perform well for workloads other than pure block IO.

CONCLUSIONS

In this paper, we evaluate the performance of the SAS Calibration Workload running on SAS 9.4 and IEEL 1.0. Our results show that having multiple mount points help resolve some of the threading bottlenecks in IEEL clients as they

currently use a single-threaded RPC to communicate with the MDS. In addition, IEEL scales well with additional resources such as more network bandwidth or more client nodes.

In conclusion, IEEL provides fast, massively scalable, cost effective and efficient storage solutions for different SAS usage models.

ACKNOWLEDGEMENTS

We would like to acknowledge Ken Gahagan, and Shane Kennedy from SAS, and Gabriele Paciucci, Arakere Ramesh, Dan Ferber, John Skerchock, and Michael Jendrysik from Intel, for their assistance and contributions to this study.

REFERENCES

[1] Lind, Bobbie, “Lustre* Tuning Parameters”, Presentation at Intel, April 16, 2013, Available at http://www.opensfs.org/wp-content/uploads/2013/04/LIND_LUG_2013.pdf

[2] Intel, “Intel® Solutions for Lustre* software”, Web link, Available at http://www.intel.com/content/www/us/en/software/intel-solutions-for-lustre-software.html

[3] Lee, Brett, “A Deep Dive on Using Intel High-Performance Fabrics and Storage to Maximize Application Performance", Web link, Available at https://intel.activeevents.com/sf13/connect/sessionDetail.ww?SESSION_ID=1231&tclass=popup

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Name: Suleyman Sair
Organization: Intel Corporation
Email: suleyman.sair@intel.com