File Access Optimization with the Lustre Filesystem at Florida CMS T2

Bockjoo Kim[1], Paul Avery[1], Dimitri Bourilkov[1], Yu Fu[1]
[1]University of Florida, Gainesville, FL, U.S.A.

Abstract

One of the CMS Tier2 centers, the **Florida CMS Tier2 center**[1], has been using the Lustre filesystem[2] for its data storage backend system since 2004. Recently, the data access pattern at our site has changed greatly due to various new access methods that include file transfers through the GridFTP servers, read access from the worker nodes, and remote read access through xrootd. In order to optimize the file access performance, we have to consider all the possible access patterns and each pattern needs to be studied separately. In this presentation, we report on our work to optimize file access with the Lustre filesystem at the Florida CMS T2 using an approach based on the analyzing these access patterns.

Introduction

The data access pattern has changed greatly due to various new access methods as is illustrated in Figure. 1.
Not all clients access the filesystem simultaneously.
However, it would be best if we can identify possible IO optimization techniques and optimize the IO from the different clients.

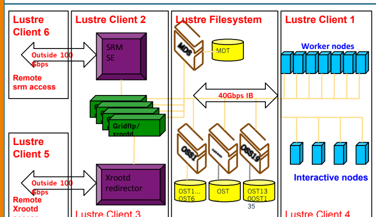
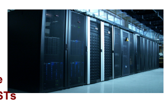
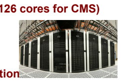


Figure 1. Lustre Filesystem, Clients, and Data Flow

Florida CMS T2 Storage

- **Bestman SE with**
  - **8 10Gbps gridftp servers**
- **Lustre Parallel Filesystem**:
  - **2.3 PB in production**
  - **85 TB under older hardware**
  - **19 OSSes with 135 RAID OSTs**
- **Serves U of Florida HiperGator[3] Imbeded worker nodes**
  - **Majority is Opteron6378@2.4GHz (4126 cores for CMS)**
  - **Torque-Moab Batch system**
- **Network:**
  - **100Gbps WAN/200Gbps Campus**
  - **55Gbps FDR Infiniband interconnection**
  - **40Gbps IB-IP bridges (among all nodes)**
  - **40Gbps NAT from worker to outside**





Performance Checks

In order to check the performance of the Lustre filesystem, the following methods are used:

- Simple Copy Test
- File Transfer Tests: Gridftp and Xrootd Test
- CMSSW IO Test

Simple Copy Test The simple copy test provides the sequential read and write rate of the the Lustre filesystem. The rate is measured as a function of the Lustre OST which is the easily testable smallest unit of the Lustre filesystem. This is the basic measurement of the Lustre I/O and tells us the baseline performance of the filesystem and the hardware . The test is performed between the Client 1 in the Figure 1 and the Lustre filesystem. Figure 2 and 3 shows the Lustre file read and write rate from the simple copy test, respectively. The copy read rate is 600MB/s for the new OSTs and 200MB/s for the older OSTs. The write rate is worse than the read rate but is similar in the OST dependence. The hard drives in the system are uniform except for one OSS where a different brand drive is used.

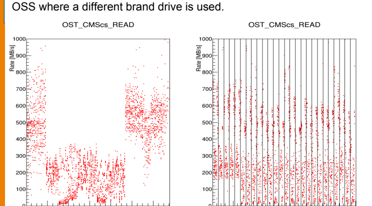


Figure 2. Lustre Read Rate from the Simple Copy Test

File Transfer Tests

The single stream transfer rate is measured within Florida T2 using globus-url-copy and xrdcp. This is to confirm the simple copy test results and the local network. The netwwork topology is illustrated in Figure 4.

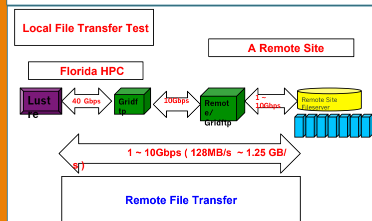


Figure 4. Network Topology in a Single Stream Transfers

Figures 5 and 6 shows the Lustre read rate using the globus-url-copy[4] from Lustre to the memory and the xrdcp[5] from Lustre to the memory, respectively. We can see they have the similar IO rates as the simple copy rates shown in the Figure 4.
The single stream transfer rate is also measured between other sites and Florida T2 using the globus-url-copy or FTS. But this is limited by the other sites SE IO rate and the network.
The average single stream transfer rate with remote sites ranges between 10 and 80MB/s depending on the network status.

CMSSW Analysis IO

CMS software typically writes compressed ROOT[6] files. In order to see if there is any discernable OST dependence, we have used the CMSSW produced ROOT files to check the analysis read rate. Figure 7 shows the CMSSW analysis read rate. As is shown, the read rate is much lower than the other read rate tests performed. This is due to the fact that the ROOT files are highly compressed and most of the read time is spent during the file decompression.
The identical inputs and the software were run at other sites by carefully staging in the input files at other sites. This was to check if there is any advantage in read files from the Lustre over other filesystem, HPFS. The result shows there is no significant difference among different storage where the input files reside.
However, the file ready rate of the CMS produced ROOT files is heavily CPU bounded as is shown in Figure 8.

Optimizing IO Activities

With the different possible Lustre access tests performed separately, a simple all client activities were emulated by running the following jobs simultaneously:
• CMSSW analysis jobs at the Lustre site
• CMSSW analysis jobs at a remote site with the direct xrootd access
• Jobs that transfer files from a remote host to the Lustre site storage element
• Simple copy jobs
The CMSSW analysis jobs at the Lustre site emulates the regular analysis jobs, the CMSSW analysis jobs at a remote site with the xrootd access emulates the more ubiquitous analysis jobs these days by reading the input files from the xrootd directly, the transfer jobs emulates the CMS PhEDEx transfers by selecting various source sites randomly. These are the control background jobs for the simple copy jobs that are the jobs from which we want to measure any IO variation. In order to find the control background job that causes the significant IO variation, number of jobs in one of the three control background jobs is varied while the number of jobs in the other two control background jobs is fixed.
From the test, we find that no particular backbground job impacts the overall performance of the Lustre system. Rather the performance depended on the number of files accessed in OSTs.
To reduce the impact from such access, the uniform distribution of files among OSTs is important as non-uniform distribution of files can impact some OSTs and that affects the whole system.

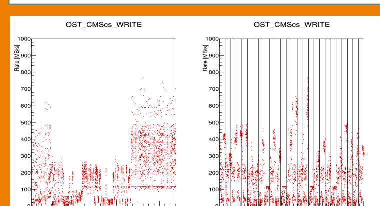


Figure 3. Lustre Write Rate from the Simple Copy Test

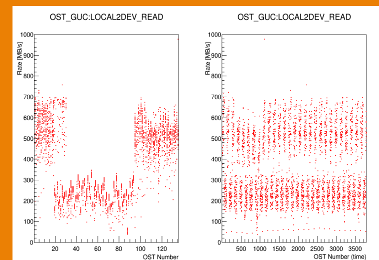


Figure 5. Lustre file read rate using the globus-url-copy

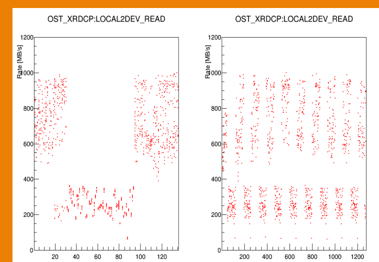


Figure 6. Lustre file read rate using the xrdcp

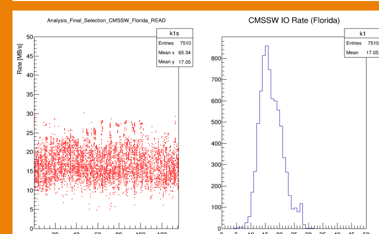


Figure 7. CMSSW Analysis OST vs Read Rate(Left) and Average Rate

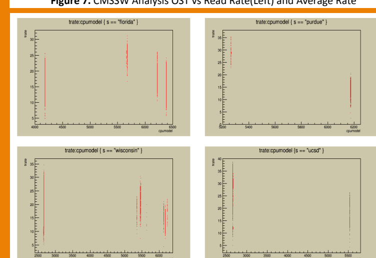


Figure 8. CPU Dependence of CMSSW Analysis Read Rate

Conclusions

We have performed IO tests with three different file access mechanism and no critical issue was found. The methodology is not limited to Lustre but can be applied to other storage systems
We find various IO can cause trouble in some IO servers, high load on gridftp but these IO troubles are network related and either they were non-standard workflows due to network issue or very IO intensive workflows.
We find no particular file access method impacts the Lustre filesystem. We also find the uniformity of the system is important and files need to be distributed among different OSTs uniformly to avoid the performance reduction. This has been implemented and run regularly to ensure there are no hot spots among OSTs.
In a very quiet condition, we were able to achieve very high sequential read and write rate and the Lustre filesystem may be used for the very high IO intensive workflows.
This has been a good learning practice and we can project the work can be extended to find the more efficient way of accessing files on Lustre.

Contact

Bockjoo Kim (bockjoo@phys.ufl.edu)
Dimitri Bourilkov (bourilkov@phys.ufl.edu)
Paul Avery (avery@phys.ufl.edu)
Yu Fu (yfu@rc.ufl.edu)

References

1) http://web.ihepa.ufl.edu
2) http://lustre.org
3) http://research.ufl.edu/publications/exploremagazine/summer-2013/hipergator.html
4) http://toolkit.globus.org/toolkit/docs/5.0/5.0.0/data/gridftp/user/
5) http://xrootd.org/
6) https://root.cern.ch/drupal/content/documentation