Tuning grid storage resources for LHC data analysis

W. Bhimji[1], J. Bland[2], P.J. Clark[1], E. G. Mouzeli[1], S. Skipsey[3] and C. J. Walker[4]

[1] University of Edinburgh, School of Physics & Astronomy, James Clerk Maxwell Building, The Kings Buildings, Mayfield Road, Edinburgh EH9 3JZ, UK
[2] University of Liverpool, Oliver Lodge Laboratory, P.O. Box 147, Oxford Street, Liverpool L69 3BX, UK
[3] University of Glasgow, Department of Physics and Astronomy, Glasgow G12 8QQ, UK
[4] Queen Mary University of London, Department of Physics, Mile End Road, London E1 4NS, UK

E-mail: wbhimji@staffmail.ed.ac.uk

Abstract. Grid Storage Resource Management (SRM) and local file-system solutions are facing significant challenges to support efficient analysis of the data now being produced at the Large Hadron Collider (LHC). We compare the performance of different storage technologies at UK grid sites examining the effects of tuning and recent improvements in the I/O patterns of experiment software. Results are presented for both live production systems and technologies not currently in widespread use. Performance is studied using tests, including real LHC data analysis, which can be used to aid sites in deploying or optimising their storage configuration.

1. Introduction

The work presented here focuses primarily on "Tier 2" WLCG sites which have limited financial resources, but are also expected to play host to a considerable portion of the user analysis of data for the LHC experiments. The large data volumes and random-like access of that data stresses such systems, hitting limits in, for example, disk seeks or network bandwidth. This results in certain setups displaying cpu efficiencies less than 50% with correspondingly low event rates. Improving these efficiencies could make considerably better use of existing computing resources, yielding savings in the substantial costs involved in provisioning LHC computing hardware. This issue needs to be addressed by the LHC community especially as we enter an era of rapidly growing data volumes and increasing analysis activity.

1.1. Tuning storage

There are a number of ways to tackle the bottlenecks and poor efficiencies described above. Some approaches are outlined below and examples of these are explored in this paper.

- **Applications:** the way data is accessed has significant impacts on the ability of the storage systems to effectively deliver it. As the data stored by LHC experiments at these sites uses almost exclusively ROOT-based [1] formats, there is scope for improvements to be made in that access layer. We examine some recent attempts to make more sophisticated use of ROOT features.

- **Filesystems/Protocols:** there are many filesystems and protocols used by LHC experiments to access data. Each can offer benefits but require specific tuning. We test in detail the RFIO protocol of the Disk Pool Manager (DPM) [2], the most widely used SRM solution in the UK, as well as exploring alternative and experimental technologies.
- **Hardware:** upgrading hardware is often the obvious way of resolving a bottleneck from the perspective of a site that may be unaware of details of the application. Current options include use of 10 Gbit networking or storage that can sustain high levels of I/Os per second (IOPS) such as Solid State Drives (SSDs). This will not be discussed in great detail in this report, but a comparison of SSDs with other solutions is performed elsewhere [3].

Section 2 of this paper describes the test setup used, while Sections 3 and 4 explore the issues described above and Section 5 considers some alternative technologies for the future.

2. Description of tests performed

2.1. Data

The tests performed here replicate LHC data analysis usage. Experimental data in the form of ATLAS “AOD” files [4] are chosen as this format is commonly used at these sites. Larger file-sizes (2-3 GB) are selected in order to make it less likely that the file can be entirely cached in RAM which would not test the underlying storage. Part of this study includes understanding how recent changes in the ATLAS data format impacts on file access performance, so the samples include identical data in older formats as well as that with the latest optimisations.

2.2. Tests

A simple framework was developed incorporating three tests:

(i) **File copy:** many ATLAS jobs currently copy data to the worker node to run. Therefore performing a file copy, using the available local file protocols, models the interaction jobs would have with the data servers and provides an easy test to run.

(ii) **Reading events in ROOT:** this test reads through the main collection tree. It provides a portable test that does not require an experiment’s software to be installed. It may however miss subtleties of real analysis jobs such as the reading of multiple trees in the same file.

(iii) **ATLAS “Athena” jobs**: we use “D3PDMaker”, which is commonly used to write data out in a different format for further analysis. This provides a realistic test of the I/O involved in reading events but requires the ATLAS software infrastructure.

In addition to this framework, use was made of the “Hammercloud” tool [5] which provides automatic resubmission of jobs together with collection of statistics. The Hammercloud tests were configured to use the same D3PDMaker job and files as above and the raw results filtered to ensure the same CPU type and similar numbers of simultaneous running jobs for each test.

2.3. Sites

Three UK Tier 2 sites were used for tests. These were:

- **Glasgow:** tier 2 site with ≈2000 CPU Cores (Intel Xeon E5420, 2.50GHz selected for these tests); ≈1PB Disk with disk servers containing 24x1TB SATA disks and connected with dual-bonded-gigabit ethernet. The storage solutions tested were DPM used in production and a DPM-Fuse test installation.
- **Edinburgh Compute and Data Facility (ECDF):** ≈1000 CPU Cores (Intel Xeon X5450, 3.00GHz selected here); ≈100TB Disk with 15x1TB SATA disks per production server and dual-bonded-gigabit ethernet. The storage solutions tested were StoRM with GPFS and DPM used in production and testbeds for the HDFS and Ceph filesystems.

- **QMUL:** ≈2000 CPU Cores (Intel Xeon, E5420, 2.50GHz tested here); ≈500TB Disk with 30x1TB SATA disks per server and 10 GbE for disk servers / dual-bonded-gigabit ethernet for worker nodes. The storage solution was StoRM with the Lustre filesystem.

Tests were run using the production infrastructure of the sites. This has the advantage of presenting studies on large scale resources that have been somewhat already tuned for LHC analysis. It represents a realistic environment that users' jobs would currently be running under. A disadvantage, however, is that some comparisons between protocols are on different site setups, though we note where that is the case. Another disadvantage is that the sites are running other jobs during the tests and to mitigate this all tests were repeated several times. The results presented should not be thought of as precise measurements but, given the large scale of some of the effects observed, definite conclusions can still be drawn.

3. Application tuning

3.1. ROOT I/O

Recently there have been a number of improvements in the LHC experiments' use of features in ROOT including basket reordering and the TTreeCache. These are described in detail elsewhere [7]. Both have the effect of reducing random-like access of data files and the requests made to the underlying filesystem and can therefore significantly improve storage performance. We measure the impacts of the changes in more detail in the following sections.

3.1.1. Basket Reordering The ATLAS experiment has recently written out files in which previously unordered buffers are arranged by event number. In Figure 1 we show the effect of this change when accessing data from the local disk (see [6] for more details on the variables plotted in this figure). Six ATLAS Athena jobs are run on different 2GB files on the same filesystem. The thick lines in the "Disk IO" plot for unordered files illustrate that the application is having to seek to various positions in the files given by the "disk offset" value. This reaches the available seek limits as shown in the "Seek Count" plot. For reading new files it can be seen that the scatter in offsets is much smaller and so seek counts are substantially reduced.

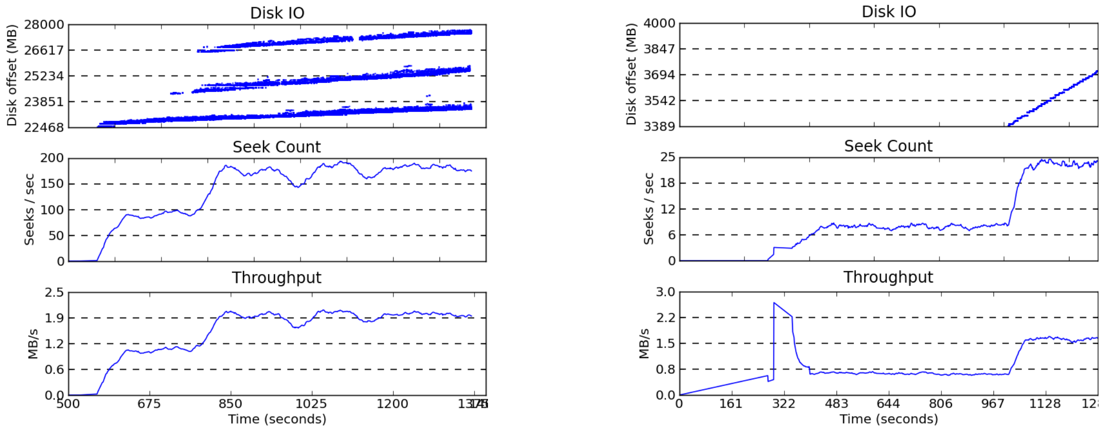

Figure 1. Disk access pattern and seeks per second for six Athena jobs running on files with unordered baskets (left) or after basket reordering (right) (plots from seekwatcher [6]).

Figure 2 shows the improvements that can be gained from reordering when data is read from remote servers using a ROOT test with different protocols at the same site. Much less time is taken to read the unordered file on the GPFS filesystem than using DPM/RFIO but the best performance for both protocols is achieved with the reordered data. Figure 3 shows larger scale

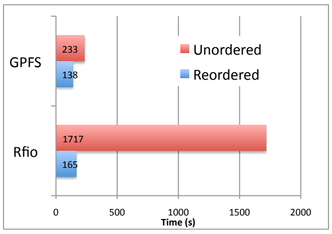

Figure 2. The time for a ROOT test reading the same unordered or reordered file for GPFS and RFIO.

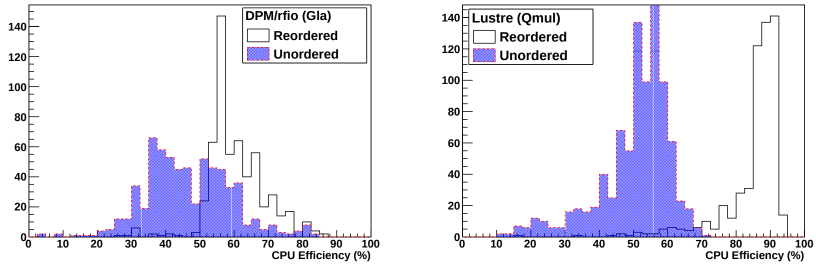

Figure 3. Stress tests running on multiple files with basket reordering (white) or without (blue, filled) on DPM/RFIO (left) (with a RFIO buffer size of 0.5M) or Lustre (right).

tests with multiple simultaneous jobs (using a “MuonAnalysis” Hammercloud test) and again there is a clear indication that the reordered files lead to significantly better cpu efficiencies for both Lustre and DPM/RFIO systems. It also appears that direct access on the Lustre filesystem offers better performance than RFIO.

3.1.2. TTreeCache The TTreeCache is a file cache for data of interest (where that data is determined from a learning phase). To use this with the RFIO protocol on DPM required fixes for a couple of minor bugs in DPM (available since version 1.7.4-7) and ROOT (backported to v5.26c, used in ATLAS releases after 15.9.0). Figure 4 shows the further improvements in event rates over basket reordering for an indicative single Athena job and Figure 5 shows the improvements in cpu efficiencies for multiple Athena jobs run simultaneously. That figure also shows that smaller RFIO buffer sizes can bring improvements in efficiency which is discussed in the next section.

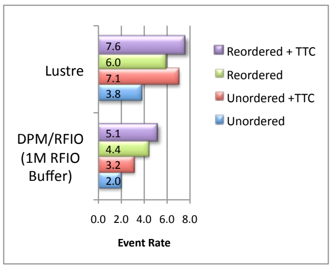

Figure 4. Event rates for an Athena test on the same unordered or reordered file with the TTreeCache (TTC).

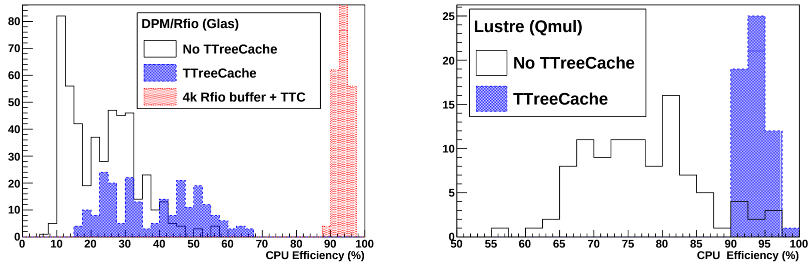

Figure 5. Cpu efficiency for multiple Athena jobs running on the same complete, reordered, dataset with the TTreeCache (blue, filled) or without (white) on DPM/RFIO (left) (with RFIO buffer sizes of 1MB and 4k (red, filled)) or Lustre (right).

4. Filesystem/Protocol tuning

4.1. RFIO buffer sizes and read-aheads

The RFIO protocol has a configurable read-ahead buffer size. Figure 5 shows that even with the reordered files, or use of the TTreeCache, higher cpu efficiencies are seen with very small buffer sizes, indicating that data access is still not sequential enough to make effective use of buffered data. Figure 6 shows the network traffic when the same ATLAS athena job is run with successively increasing buffer sizes (where linux file caches are dropped between successive runs). This illustrates that larger volumes of unused data are indeed transferred. However, using smaller buffers leads to more IOPS on disk servers and so can cause considerable load. This would impact on all other users therefore, unless they have disk servers that can handle this level of IOPS, many sites choose a higher value of the RFIO buffer size (say 0.5M) to limit the IOPS at the expense of higher network traffic. UK grid sites have also found that larger values for the block-device read-ahead on disk severs further reduces this load. Finally, it is still preferable for DPM sites to copy files to the worker node as Figure 7 also shows that high CPU efficiencies can be obtained in that case without the level of IOPS caused by direct RFIO access.

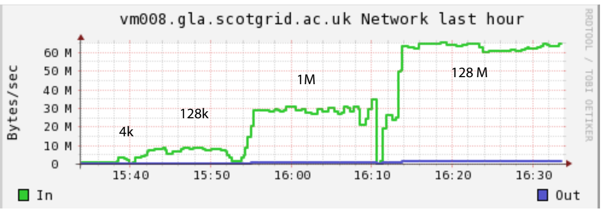

Figure 6. Network traffic for Athena tests with different values of RFIO buffer size.

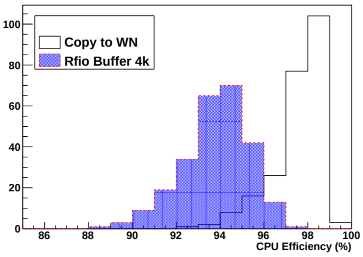

Figure 7. Cpu efficiency when the data file is copied to the worker node compared to that for direct RFIO access with a 4k buffer size.

Copying the file can however move the bottleneck to the worker node disk. As shown earlier, in Figure 1, multiple Athena jobs running on unordered data saturate the IOPS on a single disk. We measured that one Athena job running on a worker node with a single SATA disk would achieve around 90% efficiency, but this decreases to around 70% efficiency when the node had 8 cores occupied. This can be improved by using a pair of disks in a RAID 0 configuration (as is done in the test shown in Figure 7) or by using SSDs (see [3] for a detailed comparison).

5. Alternative technologies

5.1. Fuse with DPM

POSIX I/O for accessing files considerably aids users and allows use of Linux file caching. We test one way to enable this for DPM through an already existing, but not widely used, GFAL Filesystem in Userspace (FUSE) module. When using this module, RFIO is the backend protocol, but applications benefit from the page cache. As shown in Figure 8, we find that while performance for copying files is not as good as using the RFIO protocol directly, the performance for directly reading files, either in ROOT or Athena, appears to be better. However, there is currently a limitation of the module, believed to be in GFAL, that causes a crash when a file is opened twice. This impacts any Athena job with AutoConfiguration, including the D3PDMaker job used here (for testing it was resolved by opening the file via RFIO for the configuration step). It would need to be fixed for this FUSE library to be used for ATLAS Athena jobs. However, even without a fix, this module could be useful for easing access to user data.

5.2. HDFS and Ceph

A number of distributed filesystems are gaining increasing interest outside the LHC community. Among the more popular of these are the Hadoop Filesystem (HDFS) [8] and Ceph [9]. HDFS is widely used in industry and is particularly valuable for aggregating disk in worker nodes given its fault tolerance. It has already been used for LHC data in US CMS Tier 2 sites. Ceph, while not yet believed production ready by its developers, has a client that is included with the linux kernel since v2.6.34. A testbed was setup with these filesystems, full details of which are given in [10]. For ATLAS data, as shown in Figure 9, both filesystems were found to offer similar performance for reordered data as the sites GPFS and DPM systems (which were shown in Figure 2). Ceph also showed good performance for the random access required for unordered ATLAS files and while the HDFS test did not complete on those files it is believed that the default values for read-ahead in HDFS could be tuned better for this kind of access.

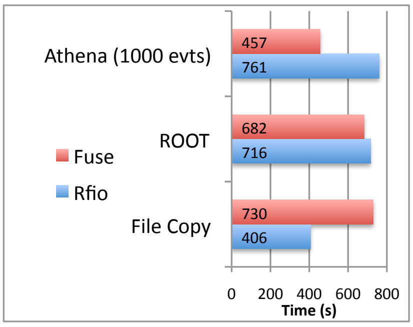

Figure 8. Time taken to complete tests reading a file directly with RFIO or the GFAL FUSE module.

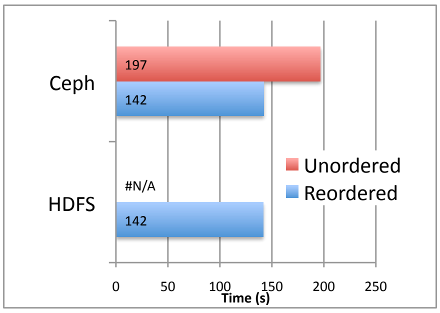

Figure 9. The time for a ROOT test reading an unordered or reordered file with the Ceph or HDFS filesystems.

6. Conclusions

Storage setups used for LHC data analysis can benefit from detailed study and tuning to improve efficiencies and event rates. Recently considerable progress has been made and we have shown that improvements in the experiments' use of ROOT I/O through Basket Reordering and the TTreeCache significantly improve performance on all the systems tested. We have also shown that tuning of those systems, such as read-aheads and the choice of protocol, can yield further benefits and that there are many promising emerging technologies. However, this tuning depends on the available site hardware and the type of jobs, so there will be a continuing need for the tools and understanding developed here to be applied through ongoing testing and monitoring.

Acknowledgments

The authors wish to thank Jens Jensen and others in the GridPP Storage Group for comments and support and Daniel van der Ster for help with Hammercloud tests.

References

[1] Brun R and Rademakers F 1997 *Nucl. Instrum. Meth.* A **389** 81
[2] DPM - Disk Pool Manager URL https://svnweb.cern.ch/trac/lcgdm/wiki/Dpm
[3] Skipsey S, Bhimji W and Kenyon M 2011 Establishing applicability of SSDs to LHC Tier-2 hardware configuration. To be published in *J. Phys.: Conf. Series*
[4] Duckeck, G *et al.* [ATLAS Collaboration] 2005 CERN-LHCC-2005-022
[5] Vanderster, D C *et al.* 2010 Functional and large-scale testing of the ATLAS distributed analysis facilities with Ganga *J. Phys.: Conf. Series* **219** 072021
[6] Seekwatcher URL http://oss.oracle.com/∼mason/seekwatcher/
[7] Vukotic, I, Bhimji, W, Biscarat, C, Brandt, G, Duckeck, G, van Gemmeren, P, Peters, A , Schaffer, R D 2010 Optimization and performance measurements of ROOT-based data formats in the ATLAS experiment. ATL-COM-SOFT-2010-081. To be published in *J. Phys.: Conf. Series*
[8] HDFS URL http://hadoop.apache.org/hdfs
[9] Ceph URL http://ceph.newdream.net/
[10] Mouzeli, E G 2010 Open Source Distributed Storage Solutions for the ATLAS Project, MSc University of Edinburgh URL http://www2.ph.ed.ac.uk/∼wbhimji/GridStorage/OpenSourceFilesystems-EM.pdf