Performance Tests of Energy Data Storage using different Distributed Parallel File Systems

WOLFGANG SÜSS, KARL-UWE STUCKY, ALEXANDER QUINTE, WILFRIED JAKOB
Institute for Applied Computer Science (IAI)
Karlsruhe Institute of Technolgy (KIT)
Kaiserstraße 12, 76131 Karlsruhe
GERMANY
wolfgang.suess@kit.edu http://www.kit.edu

Abstract: Time series captured by electrical data recording devices must be stored in efficiently stored. We specifically want to learn, in which way the large scale data can be stored. This paper describes benchmark runs using the distributed file systems iRODS, Hadoop File System and an NFS based distributed file system. For the comparison of the file systems, time series with different sizes between 5 MB and 100 MB are stored in the different file systems. The storage on each of the file systems uses Java API's. Based on the measured time for storage the file systems are compared and analysed. The results are the basis for the installation on the later production system.

Key-Words: power grids, time series, benchmark testing, energy data handling, distributed parallel file systems.

1 Introduction

The work presented in this paper has been motivated by substantial changes in the German power grid system to come in for the near future. As a consequence of the government's decision in 2011 to phase-out nuclear power, renewable energies are envisaged as a substitute: first and foremost wind power and photovoltaics. According to the rising share of renewables in the overall power market, effects due to their intermittency will become more and more important and will influence almost every aspect of the power system, e.g. transmission networks, storage solutions, operations control, electricity markets and so on.

Information technology and computer science will play a key role in this energy transition process to lead to a smart grid [1]. The Institute for Applied Computer Science of KIT is establishing a software infrastructure that combines semantic and mathematical modeling, simulation and optimisation environments, as well as measurement and data analysis. An integrated data management system is going to serve applications and tools as a large scale energy data platform for measurement results, models, and simulation data. The system deploys KIT's Large Scale Data Facility LSDF [2] and it supports SimLab Energy [3].

Among different kinds of data that must be stored and managed, time series are of considerable importance. They originate from realtime acquisition of e.g. power consumption, voltage, current, PMU data for amplitude and phase of voltage and current sinusoids with time-invariant frequency [4], etc. Furthermore, information derived from measurement data and time series data produced by simulation runs are relevant in the analysis environment.

Currently, IAI is going to implement a new power grid analysis approach based on time series that relie on EDR measurements [5]. The term EDR denotes Electrical Data Recording devices [6] that have been developed at IAI and measure three phase voltages at rates up to 25 kHz. Measurements are taken at different socket endpoints in the power grid at KIT's Campus North and they are synchronised by the GPS pulse-per-second (PPS) signal.

At typical sampling rates of 10 kHz each EDR acquires 6.5 GByte per day of raw data as well as pre-extracted features and characteristics. Since the number of EDR devices will be multiplied in the course of the project, standard sampling rates will probably rise and data handling services will simultaneously have to compensate for data acquisition from other measuring devices. On the other hand, data analysis applications access previously recorded data with data retrieval requirements, which comprise huge amounts of data to be delivered to clients or for internal data selection.

Therefore, detailed performance studies on time series storage have been started. They aim on guiding the development of new high-performance

ISBN: 978-1-61804-130-2

services that satisfy both writing tasks (for all kinds of time series acquisition in the electrical power grid) and reading tasks (for data analysis and control applications). A high scalability concerning the number of data sources and of simultaneous reading accesses of different kinds is required. This paper reports on the results of initial benchmark tests and their implications on the development of the data management system.

It is structured as follows: section II focuses on related work regarding the performance of data management and storage systems in general as well as for time series applications. The experimental setup is described in section III, explaining hardware and software issues as well as overviewing the storage systems that have been explored for this work. Benchmark test results are presented in section IV and discussed in view of data management development. Section V gives a summary and an outlook on future perfomance studies for time series data and other data types that are relevant in a power grid software environment.

2 Related Work

In principle the storage of time series is possible in two ways: storage in data bases or storage in file systems [7]. Due to the special circumstances that are dictated by the energy system applications in the field of and the storage in the LSDF, we confine our study to storage in file systems. The way SimLab applications currently access the data, suggests a blockwise storage that is best done in a file system. In [8] is the storage of quite similar data as our described and therein is also noted that the storage in a file system is most suitable.

Several articles are relevant for the work presented here. A good overview of different file systems is given in [9]. Rules for a performance measurement process are defined and discussed in [10]. Techniques for managing time series are described in [11,12]. [11] deals with reduction techniques to save storage space, but what in this paper is not yet relevant, but it will be relevant for the later production system. [12] covers different replication mechanisms that are important in this paper.

Performance tests for Hadoop [13] are discussed in [14], whereas iRODS [15] is treated in [16]. In this paper, the requirements are a little different from ours. The main focus is on the parallel transfer of many files. A defined number of files are transferred simultaneously and the runtime for the whole transfer process is measured. In our benchmark runs we measure the time for transferring one file.

3 Experimental Setup

This section describes the experimental setup of the performance test series, i.e. the file systems together with the storage APIs that runs our benchmark tests, the data generation and the construction of the benchmarks themselves.

3.1 File Systems

The Hadoop File System (HDFS):

HDFS is installed in the Large scale Data Facility LSDF, and therefore a broad range of support is guranteed. It also is the most frequently used system, several SimLabs store data into HDFS. Provided our comparison delivers good results for HDFS, many synergy effects would arise when starting to develop a data management system based on Hadoop, with extensions even in other high impact research areas such as life science, climate research, or particle physics.

The Network File System (NFS):

Similar arguments apply to NFS. Besides, it is a commonly used file system, and serves as a kind of standard helping to assess the other two systems.

The integrated Rule-Oriented Data System (iRODS):

iRODS has been selected firstly based on information from literature. According to [16], it is suitable for massively parallel requirements. Secondly, iRODS offers many features to build up a data management infrastructure, and thus may save a lot of development time, if performance is adequate.

We have installed these three file system in a test cluster in order to have comparable conditions for running benchmarks on these three file systems.

The test cluster consists of the five computers: iai-grid1 through iai-grid5. Iai-grid1 to iai-grid4 are quad core computers with an Intel dual xeon E5430 processor. Iai-grid5 is an dual core computer with an AMD-Opteron processor. The operating system is the Linux system Suse 10.3. iai-grid1 though iai-grid4 has 16 GB main memory, iai-grid5 2 GB.

On this test cluster the HadoopVersion 0.20.2+737 and iRODS Version 2.4 are installed. Iai-grid5 serves as the central node, in the hadoop terminology as name node, whereas iai-grid1 to iai-grid4 are clients, in hadoop kown as data node. In both file systems we use a replication factor of 3.

Based on NFS (Network File System) we build

ISBN: 978-1-61804-130-2

our own distributed file system. Iai-grid5 is the central point, where iai-grid1 to iai-grid4 are mounted via NFS. Implemented in Java, a distributed file system was developed that allows the storage of data over the central computer iai-grid5 to the clients iai-grid1 to iai-grid4. The replication factor can be determined by the user. For comparison with HDFS and iRODS we also use an replication factor of 3.

3.2 Data Generation

In order to get comparable performance measures for the three file systems, we use generated data. Data generation is done through a Java class and is varied from 5 MB to 100 MB in increments of 5 MB. Handled in a Java String the generated data are used for writing, reading and deleting.

3.3 Storage API

Storage on each file system is based on a Java API, which was developed for these file systems. Therefore, a Java Interface IAIStorage was specified, which defines the following methods:

```
void saveToStorage(String fn, String data);
String readFromStorage(String filename);
boolean fileExists(String filename;
boolean fileDelete(String filename);
```

For each file system a Java class,

```
<file system name>Storage.java
```

was developed. These classes implement the Java interface described above and will be used for writing, reading and deleting the generated time series on the file systems. It is important to recognize, that we use Java instead of shell commands, which would probably have been faster, but the subsequent production system will be implemented on Java for recording and storage of the energy data.

3.4 Benchmark Configuration

As mentioned above for the different benchmark runs the block size varies from 5 MB to 100 MB in step of 5 MB. Each blocks size is repeated 100 times. Time is measured by a code statement before starting and after writing, reading or deleting one block. For each block size and each repetition these two times are logged in a file. With three file systems, three different storage actions, 20 different block sizes and 100 repetitions for each block size there are 18000 measured pairs of start and end time.

For realization of the benchmark run, a java application has been developed. This Java application can be configured so that the variation of the benchmark parameters are set to appropriate Also, the file system can be easily configured. The Java application uses the storage classes discribed above.

For starting the benchmark application we use shell scripts, so that the benchmark runs can be started automatically at night, when our test cluster is not used by other applications. To compare the utilization of the cluster, we supervise the workload of our test cluster. The monitoring of the workload shows that our benchmarks are running exclusively on the five computers of the test cluster.

4 Results and Discussion

After running the benchmarks we now discuss the results.

4.1 Writing

The results for writing are shown in the Figures 1 and 2. In Figure 1 the average time for the repetitions is denoted. The x-axis shows the block sizes from 5 to 100 MB, whereas the y-axis shows the average time in milliseconds for writing data packages of the denoted block size. Figure 2 shows the average time spent for writing one MB. Comparison of the different file systems shows that iRODS is faster than the other two file systems, which require a fairly similar time to each other.

4.2 Reading

The results for reading are shown in the Figures 3 and 4. As in the figures for writing, in Figure 3 the average time for the repetitions is denoted with the x-axis showing the block sizes and the y-axis showing the average time in milliseconds for writing data packages of the denoted block size. Figure 4 shows the average time spent for reading one MB. The result is a little bit different for the writing case. NFS and iRODs are the best with quite similar performance values. HDFS is a little worse than the other two file systems.

4.3 Deleting

The last case studied is the deletion of files. The results for deletion are shown in the Figures 5 and 6 according to the previous two cases. The result is different from the writing and reading cases. Here HDSF and iRODS have the best performance, whereas NFS is a little bit worse. This probably comes from the fact that HDFS and iRODS only delete file references. The NFS-based file system holds no references to the various replications and therefore must first looking for the appropriate storage locations.

ISBN: 978-1-61804-130-2

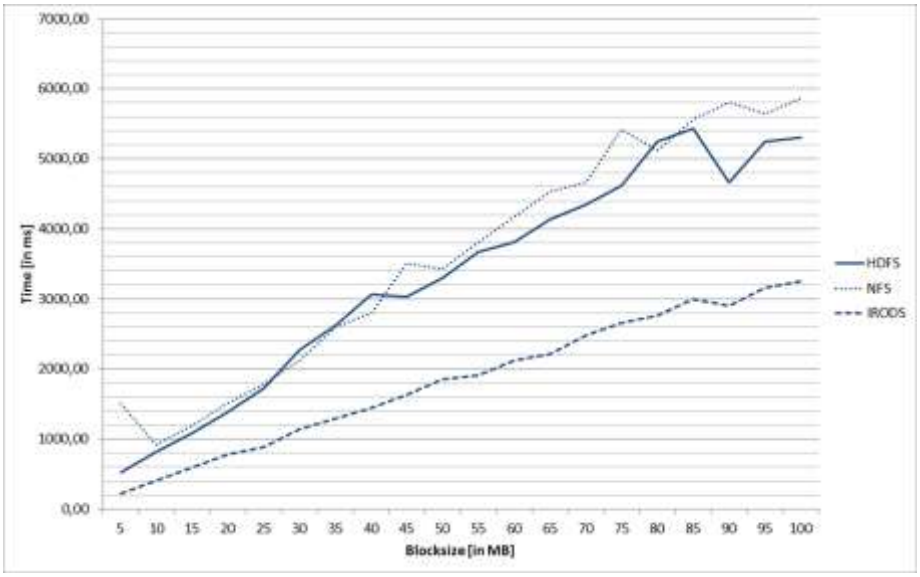

Figure 1. Average time for writing the different block sizes

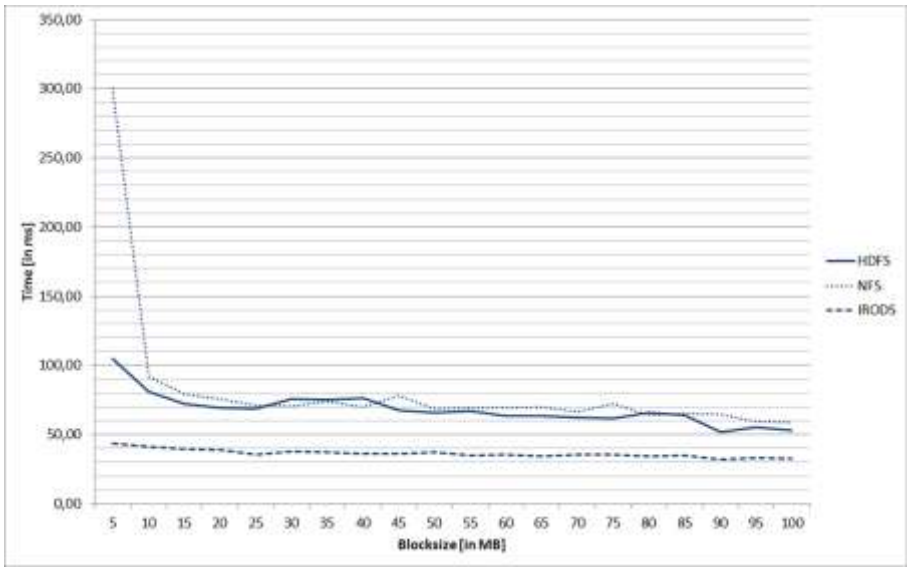

Figure 2. Average time for writing one MB

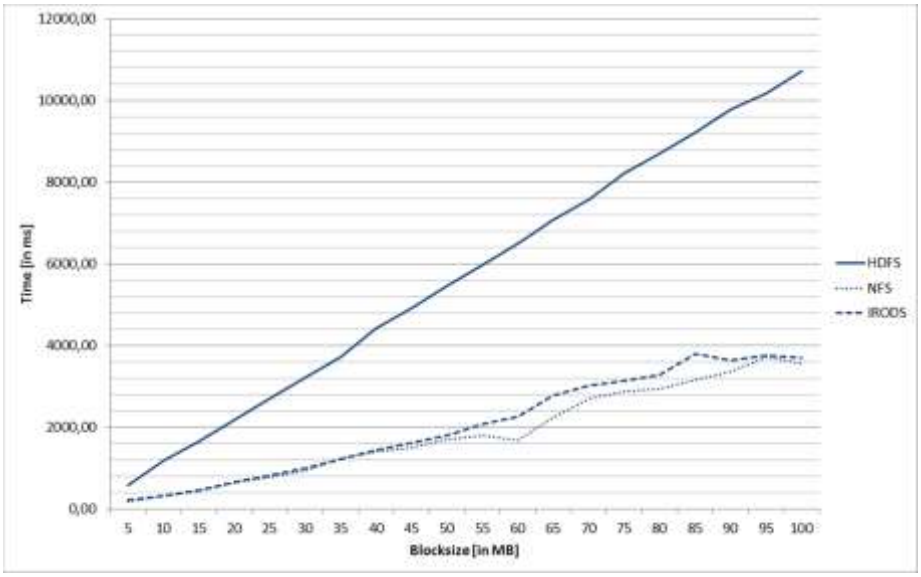

Figure 3. Average time for reading the different block sizes

ISBN: 978-1-61804-130-2

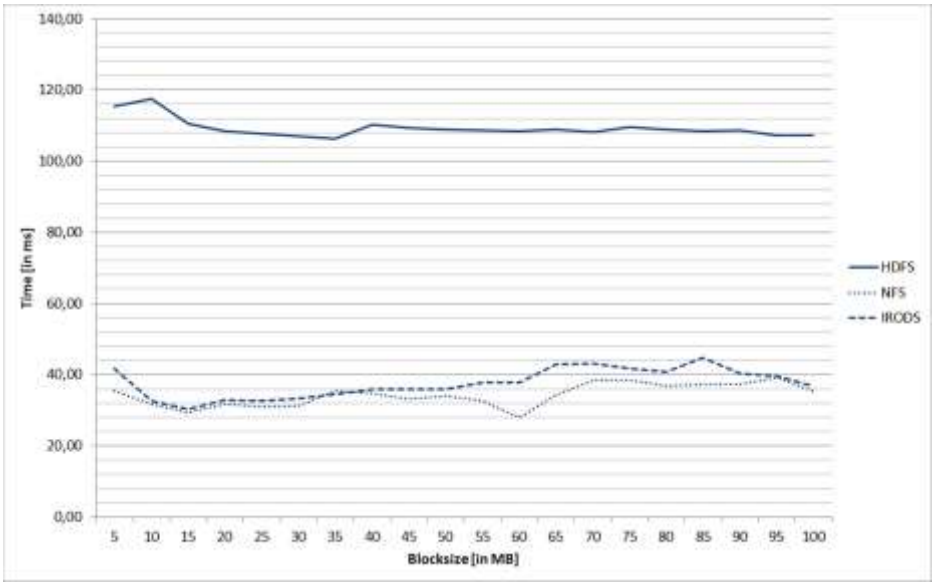

Figure 4. Average time for reading one MB

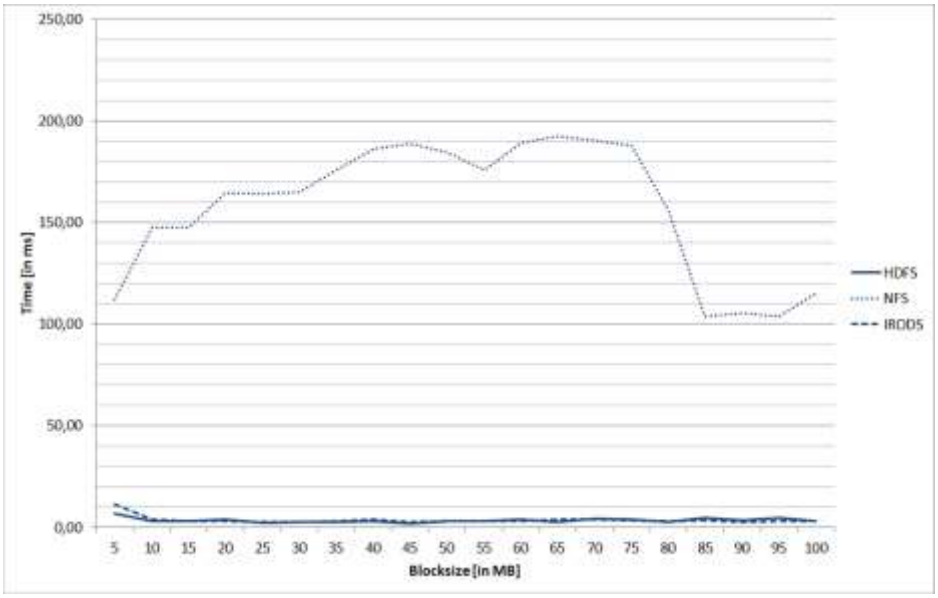

Figure 5. Average time for deleting the different block sizes

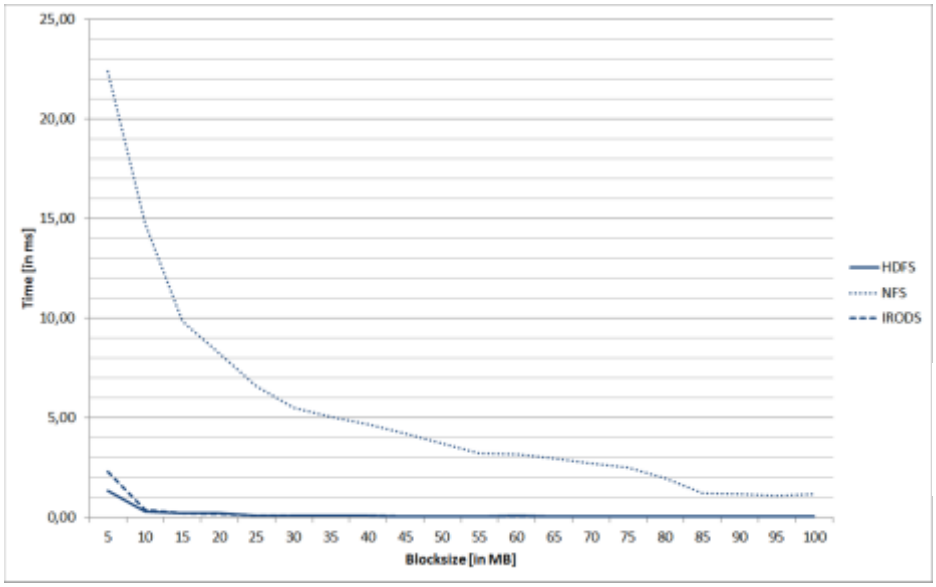

Figure 6. Average time for deleting one MB

ISBN: 978-1-61804-130-2

5 Summary and Outlook

In a first test we have compared three file systems (Hadoop, NFS, and iRODS) to learn about their performance features in storing time series data. We compared writing and reading tasks in blocks of different sizes, and also the deletion operations.
As an overall assessment we can state that all file systems are suitable.

The result will help to decide which file system will be used to build a production data management system for real data from energy data recording and also from simulated data, e.g. power plant simulations.

In the near future, the performance assessments will be extended to more specific requirements, such as accessing dedicated time intervals of single series, as well as data from multiple series for one or several instants of time. There are also further storage systems, merely promising useful features, or required by possible user groups. One is GPFS (General Parallel File System [19]) another one will certainly be Tahoe-LASF (Tahoe Least Authority File System). The latter offers security features like e.g. encryption, which are indispensible in a sensible energy application like smart meter data analysis or working with transmission line data or other economically relevant energy data sets.

A completely different approach for the storage of time series is the use of databases. As the next step we want to implement the storage of time series using databases and to compare these two different approaches of the storage of time series.

References:

[1] Amin, S.M. Wollenberg, B.F., *Toward a Smart Grid*, IEEE P&E Magazine, 3(5), pp34–41, 2005.

[2] Garcia, A.; Bourov, S.; Hammad, A.; van Wezel, J.; Neumair, B.; Streit, A.; Hartmann, V.; Jejkal, T.; Neuberger, P.; Stotzka, R., *The large scale data facility: data intensive computing for scientific experiment*, Proc.of the 25th IEEE International Parallel & Distributed Processing Symp. (IPDPS-11), Anchorage, Alaska, May 16-20, 2011 Los Alamitos, Calif. : IEEE Computer Society, 2011 S.1467-1474 ISBN 978-1-4577-1537-2

[3] Schneider, O.: SimLab Energ. In I. Kondov (Hrsg.): *Workshop Computational Methods in Science and Engineering (SimLab @ KIT)*, Karlsruhe, November 29-30, 2010, S.131-134, KIT Scientific Publ., Karlsruhe, 2011.

[4] J. De La Ree, V. Centeno, J.S. Thorp, A.G. Phadke, *Synchronized Phasor Measurement Applications in Power Systems*, IEEE Transactions on Smart Grid; Vol. 1, No. 1, June 2010.

[5] Maass, H. et. al., *Introducing a New Voltage Time Series Approach for Electrical Power Grid Analysis*, Appear in the Proceedings of Energycon 2012.

[6] Maass, H. et. al., *Introducing the Electrical Data Recorder as a New Capturing Device for Power Grid Analysis*, Appear in the Proceeding of AMPS 2012.

[7] Pavlo, A. et. A., *A Comparison of Approaches to Large-Scale Data Analysis*, Proceedings of SIGMOD'09, June 29-July 2, 2009, Providence, Rhode Island, USA.

[8] Bachoo, A.K.; van den Bergh, F.; Gazendam, A., *A comparison of data file and storage configurations for efficient temporal access of satellite image data*, SACJ43 pp. 66-74, 2009

[9] Rogier Spoor et. al., *Survey of Technologies or Wide Area Distributed Storage*, Project Report GigaPort3, June 2010.

[10] Bautista, L.; Abran, A.; April, A., *Design of a Performance Measurement Framework for Cloud Computing*, Journal of Software Engineering and Applications 2012, p.p. 69-75

[11] G. Reeves, J. Liu, S. Math, F. Zhao, *Managing Massive Time Series Streams with Multi-Scale Compressed Trickles*, Proceedings of VLDB 09, August 24-28, 2009, Lyon, France.

[12] Andreica, A.; Legrand, I. C.; Voicu, R., *Replication Mechanisms for a Distributed Time Series Storage and Retrieval Service*, Proceedings of ICAC'11, June 14-18, 2011 Karlsruhe. ACM 978-1-4503-0607-2/11/o6.

[13] http://www.hadoop.apache.org/

[14] Dutta, H., Kamil, A., Poolery, M., Sethumadhavan, S., Demme, J., *Distributed Storage of Large Scale Multidimensional Electroencephalogram Data using Hadoop and HBase, Grid and Cloud Based Data Management*, Editors Sandro Fiore and Giovanni Aloisio, Springer, 2011. ISBN 978-3-642-20044-1

[15] http://www.irods.org

[16] Hünich, A.; Müller-Pfefferkorn, R., *Managing Large Datasets with iRODS – a Performance Analysis*, Proceedings of the International Multiconference on Computer Science and Information Technology pp. 647 -654, ISBN 978-83-60810-27-9.

[17] http://www-03.ibm.com/systems/software/gpfs/

[18] https://tahoe-lafs.org/