Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 647–654
ISBN 978-83-60810-27-9
ISSN 1896-7094

Managing Large Datasets with iRODS—a Performance Analysis

Denis Hünich, Ralph Müller-Pfefferkorn
Center for Information Services and High Performance Computing (ZIH)
Technische Universität, Dresden

***Abstract*—The integrated Rule Orientated Data System (iRODS)[3] is a Grid data management system that organizes distributed data and their metadata. A Rule Engine allows a flexible definition of data storage, data access and data processing. This paper presents scenarios implemented in a benchmark tool to measure the performance of an iRODS environment as well as results of measurements with large datasets. The scenarios concentrate on data transfers, metadata transfers and stress tests. The user has the possibility to influence the scenarios to adapt them to his own use case. The results show the possibility to find bottlenecks and potential to optimize the settings of an iRODS environment.**

I. Introduction

TODAY and even more in future scientific research generates and will generate enormous amounts of data. To handle these data they are often stored in distributed locations within a Grid data management systems like the integrated Rule Orientated Data System (iRODS)[3]. iRODS allows a fast and easy access on files and their provenance. As it is used more and more in production environments and with rapidly increasing data sets it is of importance to be able to analyze and optimize the performance of an iRODS installation.

This paper describes a tool and scenarios which enable the measurement of the performance of data and metadata transfers as well as internal parameters of an iRODS system. First measurement with millions of files stored in iRODS show already the potential to optimize the performance. It is based on the benchmark tool BenchIT[1]. The aim of the development was to allow the users to easily optimize an iRODS environment. This paper is structured as follows: section 2 presents related work, section 3 gives an overview of iRODS and BenchIT, section 4 describes the investigated scenarios, section 5 shows the results of measurements and section 6 concludes this paper.

II. Related Work

There exist a variety of tools that define benchmarks and provide performance tests of I/O, e.g. IOZone [5] or IOR [4]. But these focus primarily on POSIX files systems. iRODS is not a POSIX accessible system but provides its own clients to access the storage.

iRODS is a relatively new product (version 1 released in 2008) and there are only a few performance evaluation results published, most of them focus on simple tests. Iida[7] measured the performance of transfers between two iRODS systems. Furthermore, he performed scaling tests for iCAT, concurrent tests for iCAT and a comparison of iRODS and SRB. Baquero[8] tested the performance of iRODS and Hadoop[9] and compared the results. On the iRODS website results of an ”Ingestion Test”[10] and a ”General Query Stress Test”[11] are shown. This are stress tests of the iCAT server. But a systematic performance analysis of iRODS was not done yet nor exist scenarios or a tool for users. Thus, the scenarios and the tool presented in this paper gives the user the possibility to test an iRODS environment and compare it with others.

III. Overview

A. iRODS

iRODS, developed by the Data Intensive Cyber Environment (DICE) group, organizes distributed data up to a range of petabytes. It is released with an Open-Source-Licence. iRODS stores data on heterogeneous storage systems (so-called ”iRODS server”) and the related metadata in a database on a special iRODS server named ”iCAT enabled server” (figure 1). Metadata are for example information on stored data, their locations or user defined information. A set of metadata is created and stored automatically by iRODS for every stored data - e.g. file, location or user. Additional information can be stored as user defined metadata.

An iRODS server has a Rule Engine which interprets rules. A Rule contains Micro-Services, arranged in a workflow, which process special tasks like replicating data or calculating check sums. Microservices can be provided by anyone. Additionally, a rule defines conditions to execute the Micro-Services and has a backup strategy if an error occurs. The definition of rules allows the users to configure the iRODS environment according to their requirements and is also one reason why performance testing is so important for iRODS. iRODS provides a number of clients - e.g. command line clients (the iCommands) or the iRODS explorer - as well as programming interfaces to transfer files to and from an iRODS server for example.

B. BenchIT

BenchIT[1] is a benchmark tool to analyze and optimize computer systems. It was developed at the Center for Information Services and High Performance Computing (ZIH) of the Technische Universität Dresden. BenchIT provides a set of small algorithms (kernels) to measure the performance of a system. The execution of a measurement or a set of

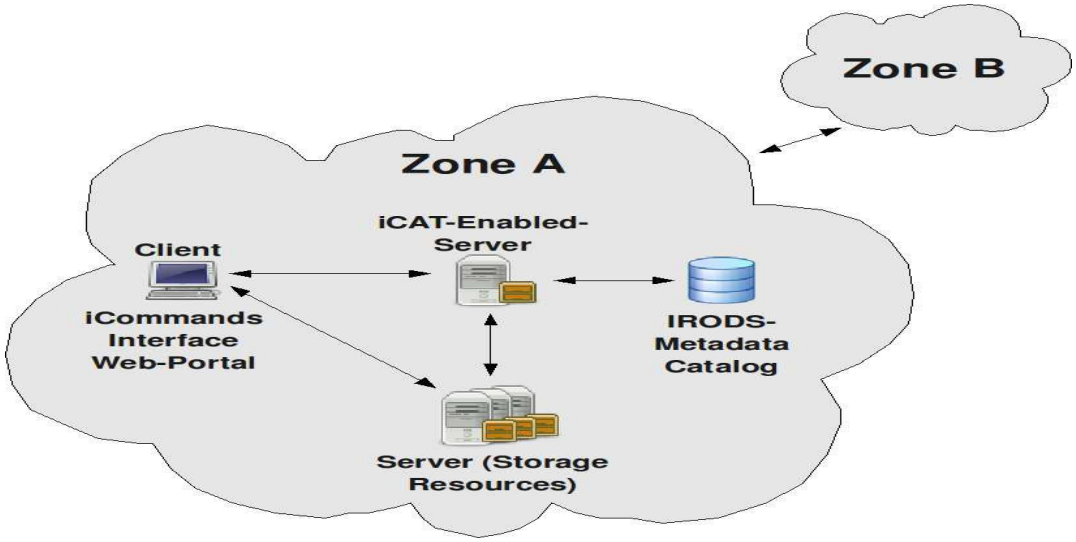

Fig. 1. Schematic representation of an iRODS environment

measurements is driven by a graphical utility providing an interface to the kernels. The interface allows the user to adjust the measurement. Relevant information on the measurement are saved together with the results in a file. The results are evaluated and diagrams of the measurement are generated. Additionally, the result file can be uploaded to a central server via a web interface [2] and can be compared with the measurements of other users.

The scenarios described in this paper were realized as kernels in BenchIT. The advantage of this approach is the use of the existing BenchIT environment with its graphical user interface and the functions to present and analyze the data.

IV. Scenarios

In the following a number of scenarios are defined that describe typical usage profiles when managing large datasets with iRODS. The scenarios are divided into four categories: data transfers with iCommands or Micro-Services, transfer of metadata, stress tests and a scenario to measure the performance of given Micro-Services. They use the client tools of iRODS such as the iCommands iput[1], iget[2], and imeta[3]. To characterize the performance the time for an event in a scenario is measured. Either the time or the derived data transfer rates are used. To avoid outliers in the results, the measurements were repeated several times and the average of the measured time were calculated.

A. Data transfer

1) Different stages of data transfer: The aim of this scenario is the measurement of different stages of a data transfer. A data transfer with iput or iget sets up the transfer environment at first. Then the connection to the iRODS server can be established. After the successful connection the client will be authenticated. If the client has the required rights the transfer starts. Usually, iRODS chooses automatically the number of parallel threads it uses for the transfer if more than one file are to be read or written. This scenario also allows a manual selection.

For the measurement a data transfer is divided in the following stages:

1) Set up of the transfer environment (Environment)
2) Establishing connection (Connection)
3) Authentication and authorization (Login)
4) Data transfer (including metadata writing) (Put File)

For these 4 stages the times are measured. Furthermore, the total time will be determined to set the single stages in relation to it. Thus, the user has the opportunity to see where the main fraction of time is used for the transfer and how it might change e.g. when increasing the number of simultaneously started data transfers. It is also possible to get an idea of how much time is used to write/read the metadata by transferring a one byte sized file because then the runtime for the last stage is mainly used for reading/writing metadata.

2) Parallel transfer of many files: This scenario transfers a defined number of files simultaneously and measures the runtime for the whole transfer process. The number of files transfered at once can be varied. In contrast to the scenario above the user gets no information about a single data transfer. The aim is to find the number of files/parallel transfers for which the iRODS environment works most efficient. The result also depends on the file size and the usable bandwidth.

3) Transfer of directories: The iCommands iput and iget provide the possibility to transfer the content of a whole directory at once. The difference between transferring a directory and transferring a number of single files is that in the first case the iRODS server only has to handle one request for the transfer and he can use all resources for this request. This

[1]iput - transfers files from the client to the iRODS server
[2]iget - transfers files from the iRODS server to the client
[3]imeta - writes metadata to the iCAT server

scenario measures the time for such a transfer with a varying number of files in the directory. The time mainly depends on the bandwidth and the time used to write metadata. The larger the files and the smaller the number of it, the more the runtime depends on the bandwidth. In the opposite case the time is mainly used to write the metadata.

4) Data transfer for varying file sizes: This scenario measures the performance of a transfer for varying file sizes. The workload for the iRODS server is small because only one file will be transfered and the iRODS server can use all resources for it. With small files the user can measure the latency and with large files the bandwidth. Furthermore the user can check the preferences set for parallel data transfers of the iRODS server for different files sizes. Therefore, it is possible to vary the number of threads for a file transfer and to compare the performance results with the results of the automatic settings.

B. Transfer of user defined metadata

To store user defined metadata on the iCAT server iRODS offers two ways. In the parallel case the metadata will be transfered with simultaneously started imeta requests each writing one element. For the performance evaluation the time to transfer all metadata is measured. In the sequential case the metadata are written in a file. The file is transfered to the iRODS server with one request, but then the metadata sets are written one by one sequentially.

The user can vary the number of metadata entries, compare the results and decide which kind of transfer is to prefer. Especially when writing large amounts of metadata these two ways can produce quite different performance results (see section V-D).

C. Stress tests

In the stress tests the iCAT server is analyzed in cases when it is flooded with requests or the number of files to store in iRODS (and thus needs to be managed by the iCAT server) increases dramatically.

1) Requests to the iCAT server: This stress test starts a defined number of parallel imeta requests to the iRODS server. The number of requests is varied to find the number of operations per second the iRODS environment can manage. Additionally, it is possible to get the number of requests the iCAT server refuse. This occurs when the maximum number of requests the database on the iCAT server can handle is exceeded.

2) Management of large metadata sets on the iCAT server: In this test the number of metadata sets the iCAT server has to manage is increased. For that, a directory with a (large) number of files is transfered successively to the iRODS environment. The time for every transfer is measured. With every file arriving in the iRODS environment the iCAT server needs to handle a larger number of metadata. This usually results in an additional time penalty when transferring the next data.

D. Time measurement of Micro-Services

Micro-Services allow the execution of small tasks directly on the iRODS server. They should not put to much workload on the server because this influences its data management capabilities. This scenario was created to measure the runtime of Micro-Services in any user defined rule. It allows developers as well as iRODS administrators to see how the performance of a Micro-Service behaves in different workflows. The scenario introduces three Micro-Services that measure the time before, between or after Micro-Services of a Rule. The results are collected at the iRODS server and moved to the BenchIT environment for analyses.

V. Results

This chapter presents measurement results of the scenarios described above. These were performed to optimize an iRODS installation to be used in production. After some generic measurements and stress tests a real use case from genetics (the future major user of the installation) was simulated.

A. The test system

The measurements were done on an iRODS installation at ZIH. The combined iRODS/iCAT server with 8 Dual Core AMD Opteron 885 (2.6 GHz) CPUs and with a memory of 32 GByte provides the storage resources and the database for the metadata over a 4 GBit/s SAN network. The client ran on a cluster with Intel Xeon Quad Core X5472 (3.00 GHz) CPUs and a 10GBits/s Ethernet network. iRODS version 2.1 was used with a PostgreSQL database for metadata storage.

B. Handling large sets of data

Figures 2 and 3 show the measurements for the single stages (see IV-A1) when 150 files are transfered simultaneously, each of them 1 Byte large. In figure 2 the iRODS environment contained no files (fresh install). Thus, the management overhead of the iCAT server is very small. Most time is used for the connection and for the file transfer including the writing of metadata. The results in figure 3 look quite different. Here, the iRODS environment already contained 13 million files and their metadata. The transfer time increased almost by a factor of 8. The reason is the increased effort to write metadata if the database is already large. This has also an effect on connection and login time. The iRODS server was not able to work off the requests as fast as before. This delay deferred the connection establishing and the login. In average the total time for the transfer increased by a factor of twenty.

C. Comparing the reading and writing of data

This test compares transfers done with iput and iget for three different states of the iRODS environment (empty, 7 million files and 13 million files stored in iRODS). A directory with a varying number of files (each with a size of 1 Byte) was written and read. Figure 4 shows the results. In the empty state reading needed 4 times longer than writing for 150 files. This means that finding the requested metadata and files needed much longer than writing both.

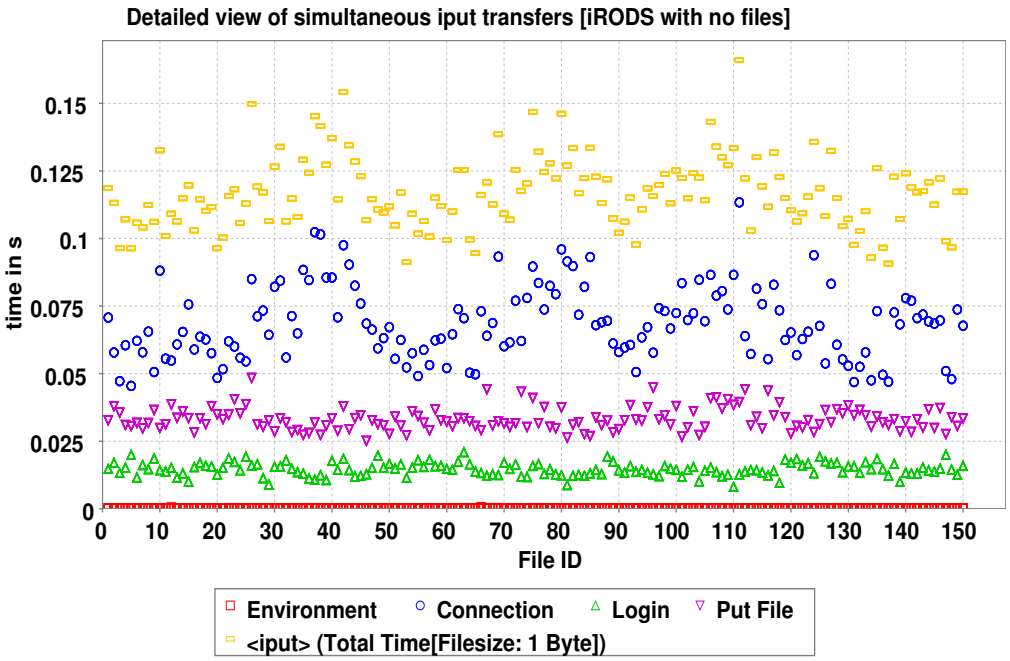

Fig. 2. Simultaneous data transfer of 1 Byte large files with iput. The iRODS environment contains no files. Shown is the time needed for every stage and the overall transfer time (stages, see IV-A1).

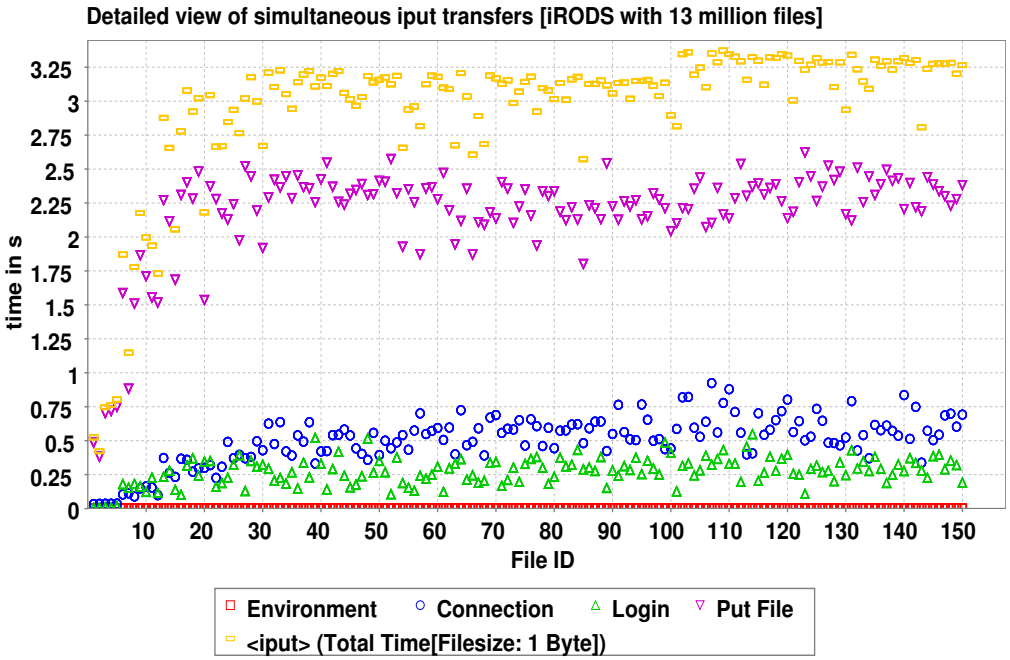

Fig. 3. Simultaneous data transfer of 1 Byte large files with iput. The iRODS environment contains 13 million files. Shown is the time needed for every stage and the overall transfer time (stages, see IV-A1).

In the next state (7 million files) the behavior changes. The time for writing data increased significantly compared to reading, where the increase is small. Nevertheless, the performance of writing was still better than of reading. The time difference between iput and iget decreased by a factor of 1.5 for 150 files written or read.

As expected, in the last case of the filled iRODS environment the performance decreased both for reading and writing. The increase of time iget needed is small compared to iput, which is 7 times slower than in the empty state. Now writing data is slower than reading files.

Summarizing one can say that the results of this test show that writing the standard metadata is an important performance factor and that the number of metadata stored on the iCAT server has more influence on writing than on reading files.

D. Writing user defined metadata

This test wrote the same amount of metadata to iRODS using the two transfer methods described in section IV-B. Furthermore, the measurements were performed with the iRODS environment in two states: empty and filled with 13 million files. Figure 5 shows the results.

In the case of the empty iRODS environment the difference between the performance of simultaneously and sequentially written metadata is small. This changed when the iRODS environment was filled with 13 million files. While the time of simultaneously written metadata increased already significantly (e.g. for a number of 150 metadata elements by a factor of 5), the performance of the sequential method got even worse (e.g. 24 times longer for 150 elements). The reason for the latter is the accumulation of the additional time (originating from the slowdown due to the filled iCAT server) needed for the sequential writing of the metadata (see description in IV-B).

E. Stress tests

On the iCAT server it is possible to configure the maximum number of requests allowed to the database. With scenario IV-C1 it is possible to measure the actual number of requests an iCAT server can handle. Measurements were done for three states - setting the maximum number of simultaneous requests to 200, 700 and 1000, respectively (the default value is 100). Additionally, the shared memory of the database was increased to 2000 MByte - otherwise the database had refused requests. The test varied the number of simultaneously started processes. Each process starts 10 imeta requests. Figure 6 shows the results for the three stages. Up to 100 processes iRODS can handle more than 35 operations per second. Between 100 and 500 processes iRODS finished the requests with about 34 op/sec. This value decreases faster, down to 27 Op/sec.

The second test (see IV-C2) continuously transfers 1000 files at once (each file with a size of 100 Byte) to the iRODS server. With every run the number of metadata to be managed increases. In the empty iRODS environment the transfer time is about 7 seconds and up to 150 files can be written per second (figure 7). With 13 million files already stored on the iRODS server, the time increases to 38 seconds and only 25 files can be transfered per second. The performance decrease is a factor of about 6. The major cause is the metadata handling in the database. The shift of the transfer time between 7 million and 10 million files was probably caused by a process on the iRODS server. Because the overall time of the measurement needed more than one day it was not possible to identify the exact reason for the shift when the measurement was finished.

F. Using the scenarios to simulate a real use case

In the following it is described how the benchmark tool and the scenarios were used to simulate a real use case. In genetics automatic microscopes take a large number of pictures of gene screening processes. For a gene screen ten thousands of pictures needs to be analyzed. The management of the pictures and their metadata is done with iRODS.

To find optimal usage parameters for this use case it was simulated using 10,000 files each with a size of 10 MByte. The transfer was done with one iput request. Figure 8 shows the time of the transfer as function of the different number of threads (parallel transfers) used and for different conditions of the iRODS environment (empty, 7 million files in iRODS and 13 million files in iRODS). The condition of the iRODS environment had more influence on the performance than the used number of threads. For such a large number of files to transfer the performance variation between the thread The difference of the writing time between an empty database and a database filled with metadata for 13 million files is about 20%.

Figure 9 shows how time and bandwidth will change if the files are stored in several directories and these are simultaneously transfered to the iRODS environment. The use of 5 directories instead of one reduces the time by more than the half and when using 10 directories the runtime is about a third. The reason is the better utilization of the bandwidth. That means in this case it is useful to divide the files on more than one directory by using subdirectories. The subdirectories can be transfered simultaneously and reduce the total transfer time to optimize the use of the bandwidth.

VI. Conclusion

In this paper scenarios for iRODS performance measurements were presented, which are integrated in the tool BenchIT. This allows administrators or users of iRODS to measure the performance of an iRODS installation easily and to adopt the usage patterns to an optimal performance. The scenarios provide performance measurements of data transfers, user defined metadata transfers, stress tests of the iCAT server and the runtime of Micro-Services. Furthermore, performance tests done on an existing iRODS environment and simulating a real use case were presented. Among other things the results show that the performance of data transfers decreases when the number of metadata the iCAT server has to manage grows significantly. But the measurement described in section (V-F) also shows that it is possible to optimize the performance by efficiently using the iRODS capabilities.

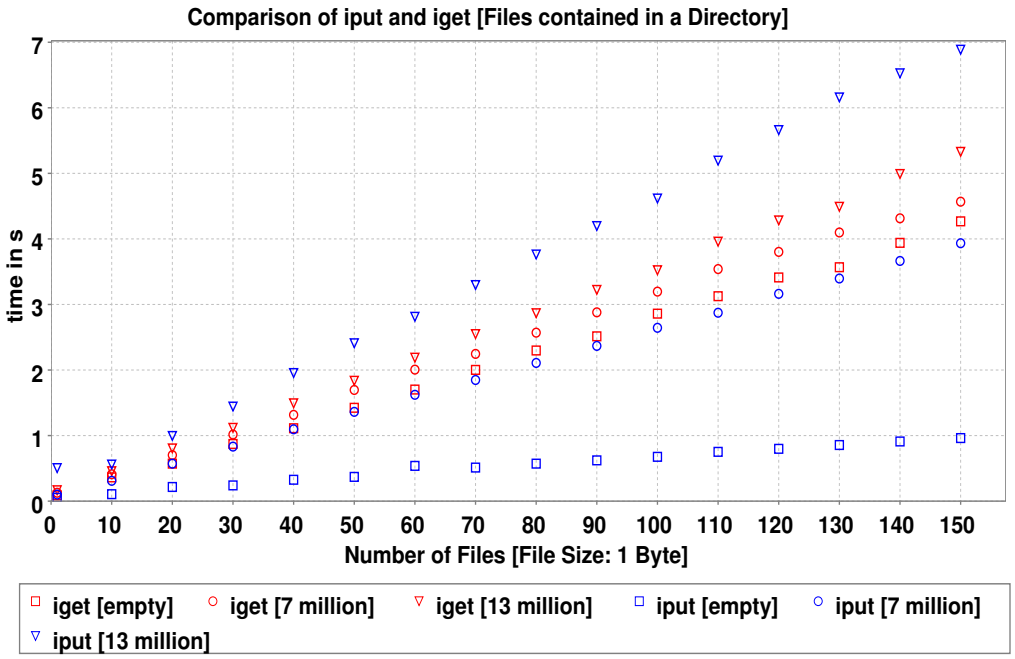

Fig. 4. Comparison of writing (iput) and reading (iget) a varying number of files (each with a size of 1 Byte) transfered at once in a directory. The figure shows the time needed for the transfers with iput and iget for three different states of the iRODS environment (no [empty], 7 million files and 13 million files already stored in iRODS).

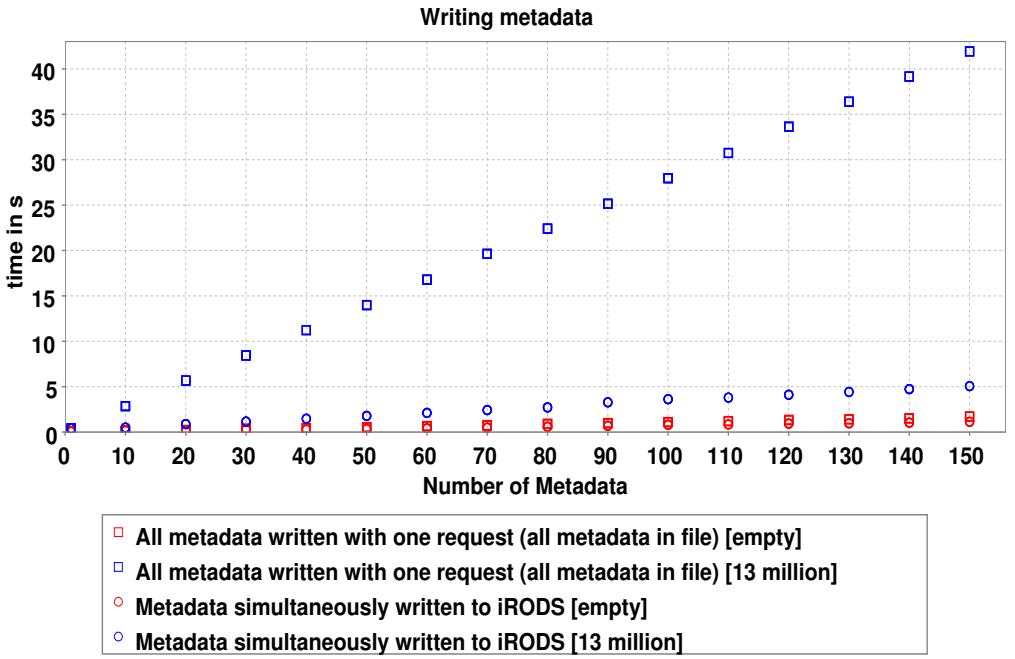

Fig. 5. Comparison of simultaneously (parallel imeta requests) and sequentially (in one file) written metadata. Shown is the time needed for writing a varying number of metadata for two states of the iRODS environment (empty and filled with 13 million files).

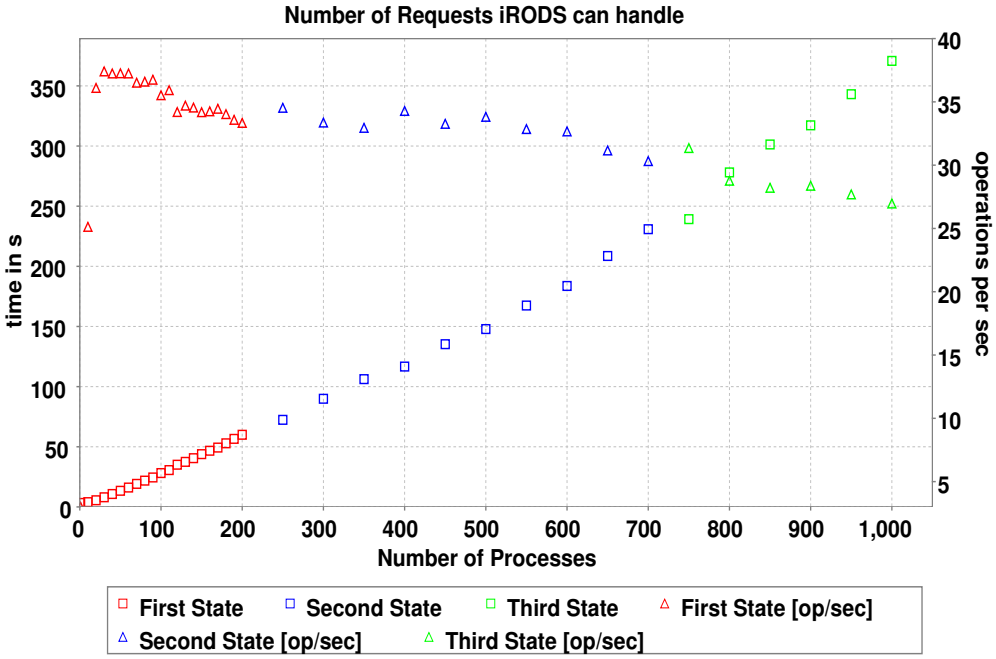

Fig. 6. Number of metadata requests: A varying number of processes were simultaneously started. A process starts sequentially 10 imeta requests. Shown are the time needed to process all requests and the resulting operations per second (op/sec). The three states correspond to the setting of the maximum number of requests allowed on the iCAT server (200, 700 and 1000).

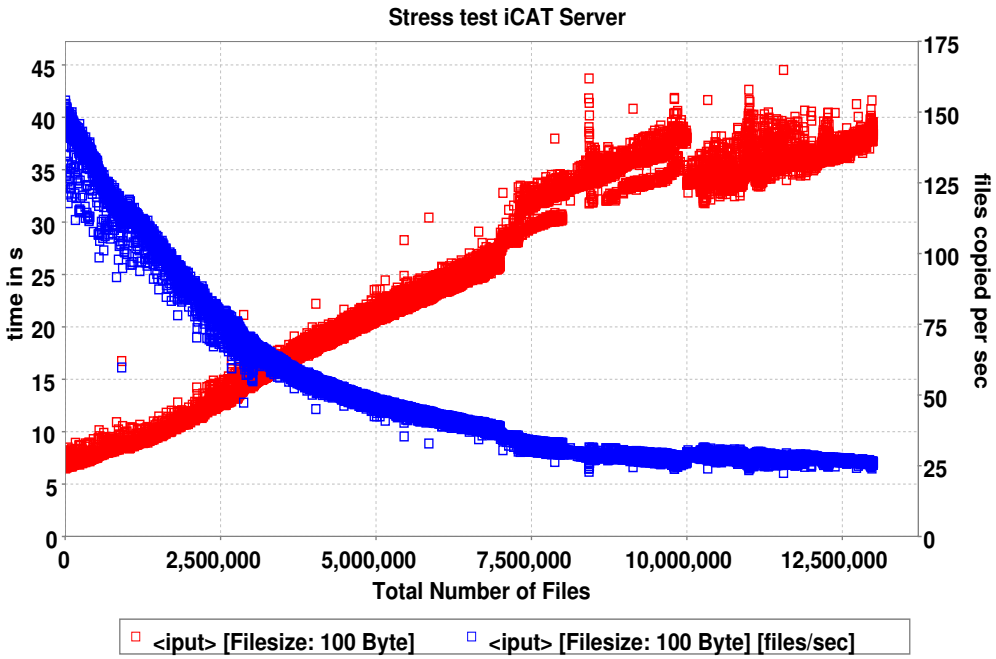

Fig. 7. Stress test: 13 million files (each 100 Byte) are transfered with iput. With each run 1000 files are transfered. Shown are the time for each run and the number of file transfers per second.

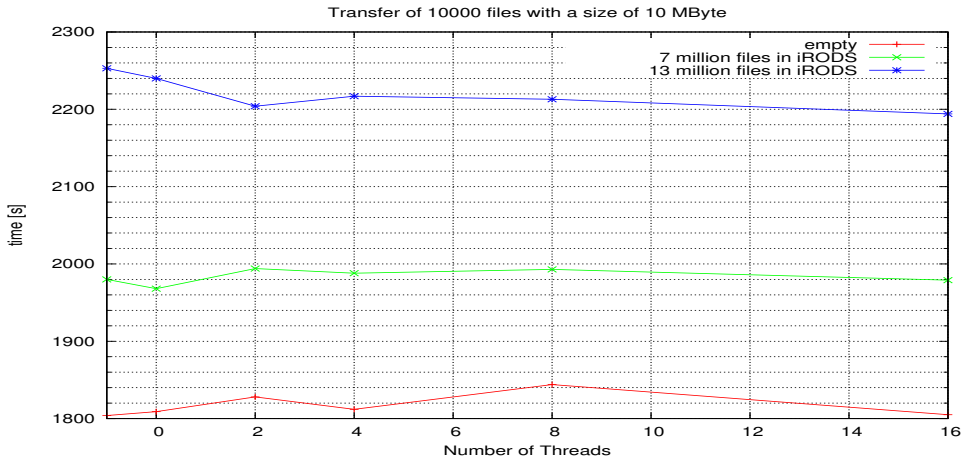

Fig. 8. Use case: 10,000 files (each 10 MByte) were written as part of one directory to the iRODS environment. Shown is the time needed for the transfer as a function of the threads/parallel transfers used. The thread number -1 means automatic thread choice by iRODS and 0 means no threading.

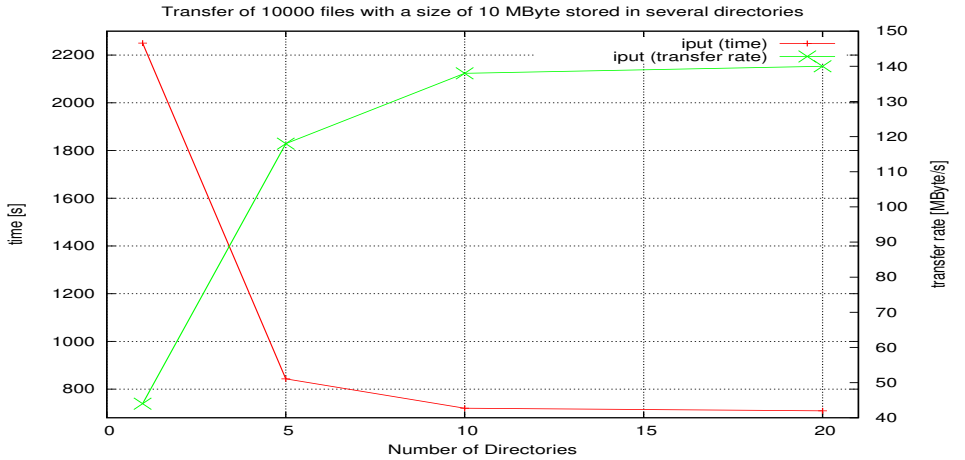

Fig. 9. Use case: 10,000 files are stored in several directories and are then written in parallel. Shown is the time needed for the transfer of the directories containing the 10000 files.

REFERENCES

[1] Guido Juckeland and Stefan Börner and Michael Kluge and Sebastian Kölling and Wolfgang E. Nagel and Stefan Pflüger and Heike Röding and Stephan Seidl and Thomas William and Robert Wloch: BenchIT - Performance Measurement and Comparison for Scientific Applications. Parallel Computing: Software Technology, Algorithms, Architectures and Aplications, Elsevire Science, The Netherlands, pp 501-508, 2004

[2] http://www.benchit.org/

[3] Reagan Moore and Arcot Rajasekar: IRODS: Integrated Rule-Oriented Data System. White Paper: IRODS: Integrated Rule-Oriented Data System (2008)

[4] H. Shan and J. Shalf: Using IOR to analyze the I/O performance of HPC platforms Cray Users Group Meeting (CUG) 2007, Seattle, Washington, May 7-10, 2007

[5] Ben Martin: IOzone for filesystem performance benchmarking Linux.com, http://www.linux.com/archive/feature/139744, Retrieved 2009-10-15.

[6] Denis Hünich: Grid-Datenmanagement mit iRODS - Entwicklung von Komponenten zur Performance-Analyse. Diploma thesis, 2009

[7] Yoshimi Iida: iRODS perfomance and KEK, UK e-Science workshop "Building data grids with iRODS", NeSC Edinburgh, 27 May - 30 May 2008.

[8] Cesar Augusto Sanchez Baquero: Performance comparison between iRODS and Hadoop Distributed File Systems, Universidad Nacional de Colombia, 15.June 2009

[9] HADOOP Website: The Hadoop Distributed File System (HDFS) http://hadoop.apache.org/common/docs/current/hdfs_design.html

[10] iRODS Website: Ingestion Testing, https://www.irods.org/index.php/iCATStressTest_at_SDSC

[11] iRODS Website: General Query Stress Testing, https://www.irods.org/index.php/icatGenQueryStressTest_at_UMIACS