Apache Hadoop as a storage backend for Fedora Commons

Frank Asseg, Matthias Razum, Matthias Hahn

Fiz Karlsruhe, Hermann-von-Helmholtz-Platz 1, 76344 Eggenstein-Leopoldshafen
frank.asseg@fiz-karlsruhe.de

Introduction

Certain types of repositories are constantly growing in size. This is true for archives, national libraries, and research institutions. Research itself is increasingly data-driven (Hey & Trefethen, 2003). This leads to vast amounts of raw and preprocessed data. Web archiving, as done by e.g. the Internet Memory Foundation[1], requires the ingestion of tens of thousands of files on a daily basis. Aside from the traditional text based publications, there is a trend to archive content like video or audio in a library. This leads to large scale data repositories posing new challenges for digital preservation tasks in terms of performance. An example for a common preservation task is the calculation of check sums on a regular basis for data degradation discovery. Running this task on a petabyte scale video archive can take more time than the interval in between scheduled executions of the task, as defined by a institutions preservation policy. Traditional repository architectures do not meet the requirements for such situations very well.

The SCAPE Project

The SCAPE project[2] is an EU-funded FP7 digital preservation project with sixteen academic and commercial partners from eight countries taking part. SCAPE will develop scalable services for planning and execution of institutional preservation strategies on an open source platform that orchestrates semi-automated work flows for large-scale, heterogeneous collections of complex digital objects. SCAPE will improve the state of the art of digital preservation in three ways: by developing an infrastructure and tools for scalable preservation actions; by providing a framework for automated, quality-assured preservation work flows and by integrating these components with a policy-based preservation planning and watch system. The achievement of these project goals will be validated within three large-scale Testbeds from diverse application areas.

Akubra Implementation for Apache Hadoop

The preservation community has to adapt to what has been coined the data deluge[3]: A logistic growth[4] of available information as a consequence of the increasing size of single entities like images and the increase in the amount of content creators. The eScience community for example has to employ new modes of data analysis in order to handle the amount of data resulting from modern experiments since the sheer volume of output data renders sequential processing impractical. Similarly in digital media archiving the requirements in regard to volume have increased proportional to the rate of growth of content published. This increase in volume in turn raises the performance requirements for processing of data in the context of curation, preservation and dissemination.

Several partners within the SCAPE consortium have built repositories based on Fedora (Lagoze, Payette, Shin, & Wilper, 2005). Examples are DOMS from Statsbiblioteket in Denmark, RODA (Barbedo, et al., 2009) from Keep Solution in Portugal, and eSciDoc (Razum, Schwichtenberg, Wagner, & Hoppe, 2009) from the Max Planck Society and the FIZ Karlsruhe in Germany. While all these systems differ substantially in their design and implementation, they only allow sequential access to digital objects and their contents, due to the current Fedora architecture and API. The idea was to push CPU and I/O-intensive operations into the

storage layer, thus enabling parallel and efficient access to the contents of the repository. Apache Hadoop has been chosen for SCAPE as the technical platform for implementing preservation actions. Hadoop clusters with lots of independent disks for storage/replication and the Map-Reduce framework for computational tasks are a good fit for the requirements of the SCAPE project and the idea of a storage layer for Fedora, that is capable of running preservation tasks in a distributed manner.
Typical examples for operations becoming impractical when executed in a sequential fashion on large corpora of data are consistency checks, format migrations and full text searches. Since such operations over the whole data set can often be broken down into smaller units of work dealing with single entities, parallelization is an effective way to increase the throughput of such processes.

Apache Hadoop is a framework for parallelization of data processing via Map-Reduce and the distributed Hadoop file system HDFS[5], based on two papers released by Google describing their Map-Reduce framework and the Google File System. Hadoop provides the user with the software system to efficiently distribute units of work onto a number of connected nodes with respect to data locality, so the need for copy cycles in between nodes of the system is minimized. Since 2006 Hadoop has become very widely used and the active community surrounding this project make it a viable choice for building a long-term preservation system, as good support for Hadoop exists in form of resources on the web as well as developers familiar with the framework. The hardware employed for a Hadoop clusters normally consists of x86 based off-the-shelf machines. One of the advantages of using a cluster of commodity machines rather than high-end computers for storage and processing lie in the ability to change the number and type of nodes in the system according to he current requirements on thy fly. This lets institutions react quickly and in a cost effective way to changes in hardware and computational requirements, necessary for e.g. migrations.

Akubra[6] is the module in the Fedora Commons repository system responsible for writing digital object representations and managed Datastreams in a persistent storage. By exposing an API developers were enabled to create implementations of Akubra for arbitrary storage systems like iRODS[7]. In Fedora's architecture Akubra is located as an intermittent layer of abstraction in between the management module and the actual storage system, which in the default configuration is a POSIX file system. The management module calls on the Akubra implementation for persisting and retrieval of serialized Fedora Objects as well as the binary content of managed Datastreams.

By implementing the Akubra API for the Hadoop file system HDFS data processing within the Hadoop framework via Map-Reduce becomes feasible without exporting the whole corpora for processing beforehand in order to perform compute intensive tasks in distributed way. The implementation consists of Java classes which use the well documented Hadoop Java API for handling Fedora’s Objects and Datastreams on the distributed Hadoop file system. It encompasses methods for reading and writing from and to files, iterating over directories for content discovery as well as the facility to map Fedora IDs to HDFS paths.

One of the challenges encountered while implementing the Akubra API for HDFS is Hadoop’s so called “small files problem”, which is directly related to the architecture of the framework and HDFS. Files in HDFS are organized in blocks of a set size, so that files larger than this threshold get split up in chunks of the set size. The Namenode in a HDFS file system is a privileged machine in the cluster managing the access to files in the distributed environment. One side of the problem is that every file, directory and block takes up a certain amount of memory in the Namenode. This sets constraints on the number of files that can be effectively managed by HDFS dependant on the memory available to the Namenode. Since having small files in a large data set implies having lots of them memory depletion and disk swapping on the Namenode slows down file system operations. The other side of the problem is the Mapping step of the Map-Reduce algorithm is normally done on the block level of files. As these blocks become increasingly small and numerous the number of individual Map tasks increases to a point where the input for a Map task is too small to be efficiently processed in comparison to the overhead introduced for the necessary book keeping of the task. This has the potential to slow down Map-Reduce jobs to a point where sequential processing would have achieved a higher throughput. We examine different possibilities to mitigate the "small files problem" in the HDFS implementation for Akubra such as Hadoop archives, Sequence files or HBase tables and their corresponding advantages and disadvantages.

Since Akubra is only responsible for persisting serialized objects and their Datastreams and not for updating indices or database information required by Fedora, only the storage layer can profit from the possibility of distribution via the Hadoop framework. Other resources Fedora depends upon for it's services like the database or the web application itself remain non scalable. In order to be able to turn Fedora itself in a distributed system another layer of abstraction in Fedora's management module as described by the High level Storage[8] proposal is needed.

Performance measurements with SCAPE test bed data on a terabyte scale have been conducted to determine the overhead of ingest and update operations compared to a standard installation of Fedora operating on a POSIX file system. Different configurations of Hadoop with respect to file size were used to be able to determine the impact of the "small files problem" on the performance of the Akubra implementation on top of the Hadoop file system. We will present these findings to give users interested in applying such a storage strategy to their Fedora based repositories an idea of the benefits gained by being able to parallelize curation tasks on a Hadoop cluster in relation to the house keeping overhead in a distributed storage.

References

Barbedo, F., Castro, R., Corujo, L., Faria, L.s., Ferreira, M., Henriques, C., et al. (2009). RODA - A Service-Oriented Repository to Preserve Authentic Digital Objects. 4th International Conference on Open Repositories. Atlanta, GA, USA: Georgia Institute of Technology.

Dean, J., & Ghemawat, S. (2004). MapReduce: Simplified Data Processing on Large Clusters. OSDI'04: Sixth Symposium on Operating System Design and Implementation. San Francisco, CA, USA.

Ghemawat, S., Gobioff, H., & Leung, S.-T. (2003). The Google File System. 19th ACM Symposium on Operating Systems Principles. Lake George, NY, USA.

Hey, T., & Trefethen, A. (2003, May 30). The Data Deluge: An e-Science Perspective. Grid Computing, pp. 809-824.

Lagoze, C., Payette, S., Shin, E., & Wilper, C. (2005, December 29). Fedora: an architecture for complex objects and their relationships. International Journal on Digital Libraries, 6(2), pp. 124-138.

Rajasekar, A., Moore, R., Hou, C.-Y., Lee, C. A., Marciano, R., de Torcy, A., et al. (2010). iRODS Primer: Integrated Rule-Oriented Data System. Synthesis Lectures on Information Concepts, Retrieval, and Services, 2(1), S. 1-143.

Razum, M., Schwichtenberg, F., Wagner, S., & Hoppe, M. (2009). eSciDoc Infrastructure: A Fedora-Based e-Research Framework. In M. Agosti et al., ECDL 2009, LNCS 5714 (pp. 227-238). Springer.

[1] http://internetmemory.org/

[2] http://www.scape-project.eu/

[3] http://spectrum.ieee.org/at-work/innovation/the-coming-data-deluge

[4] http://en.wikipedia.org/wiki/Logistic_function

[5] http://hadoop.apache.org/hdfs

[6] https://wiki.duraspace.org/display/AKUBRA/Akubra+Project

[7] https://www.irods.org

[8] https://wiki.duraspace.org/display/FCREPO/High+Level+Storage