Intermediate Gateway Service to Aggregate and Cache the I/O Operations into Distributed Storage Repositories

Mehmet Balman, Ismail Akturk, and Tevfik Kosar
Department of Computer Science, Louisiana State University
{balman, akturk, kosar}@cct.lsu.edu

Today's large-scale scientific applications generate tremendous amount of data. In addition to immense computational requirements, data management is one of the biggest challenges. Those data intensive applications usually make use of scratch volumes, which are limited in size, to store output data temporarily in a high performance computational cluster. A general scenario is to use an intermediate storage area and then transfer files to a remote storage for post processing and long term archival. Data Grids provide a distributed environment for indexing and storing large scale scientific data for collaborative science. Staging data in a fast storage space enables fast I/O access. Data generated by a data intensive application is intended to be uploaded to a specific data resource which is usually a remote system. On the other hand, we need workflow managers to organize input data retrieval, job execution, and uploading of output data. This leads to two major problems. First, there is need to handle data-flow with some external tools; second, client is limited by the storage capacity of the intermediate staging area.

We have developed a distributed data storage system, PetaShare[1], that span multiple institutions across Louisiana. PetaShare resources among the LONI[2] sites are shown in Figure 1. Petashare provides a unified namespace using iRODS[3] services and lightweight client tools for efficient and transparent access. Petashell, based on Parrot[4], provides an interactive shell interface by capturing I/O calls and matching them to corresponding remote I/O operations. Petafs, a FUSE[5] client, allows users to access remote data repositories mounted as a local filesystem. In order to overcome latency problem, we have optimized Petafs and Petashell clients by aggregating I/O requests to minimize the number of network messages. We have implemented prefetching for read operations, and caching for write operations such that we delay I/O operations and upload data in large size of chunks to eliminate the effect of high latency between the client and the data resource. Figure 3 presents some test results for advance buffer implementation in PetaShare clients.

By extending this idea to provide a transparent and efficient direct access to data repositories, we also plan to provide more intelligent FUSE clients especially for HPC applications. There is a tremendous effort to make efficient data access in cluster file systems. Our focus is to provide a transparent access to remote data repositories. However, number of compute elements involved in a large scale application is far beyond the ability a data management service can handle. Besides, multi-core compute elements in which every core is able to generate I/O calls, makes I/O latency a very crucial problem in today's high performance research. Thus, we intend to make client tools accessing remote data storage more intelligently and uploading data transparently while preserving the performance and efficiency. Instead of sending I/O request directly to the remote data resource, we plan to use an intermediate gateway service to aggregate and cache I/O operations. This will provide an asynchronous mechanism and also make I/O accesses efficient since we will not be dealing with high latency for many many small data chunks sent for I/O calls. Those clients communicate and forward I/O request to the gateway service. Gateway act as a staging area, but transparent to users. It does caching and aggregation of I/O requests. We sent data in large chunks and minimize number of calls sent to the remote data repository. This model works for most cases since HPC applications usually do not necessitate complex I/O patterns. Figure 2 gives a better representation of the intended system architecture.

1 PetaShare: http://www.petashare.org
2 Louisiana Optical Network Initiative: www.loni.org
3 iRODS: http://www.irods.org
4 Parrot: http://www.cse.nd.edu/~ccl/software/parrot
5 FUSE: http://fuse.sourceforge.net/

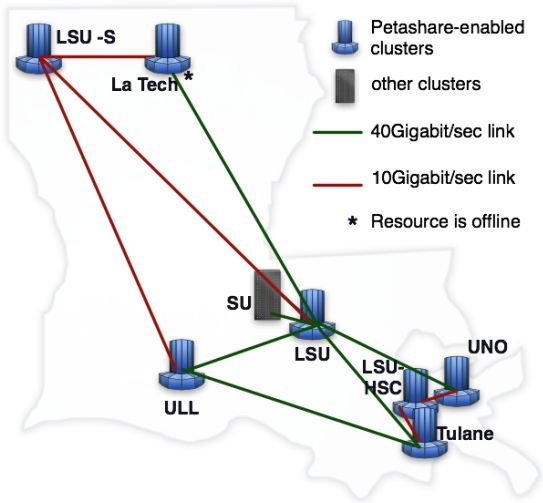

Figure1: LONI and PetaShare Sites

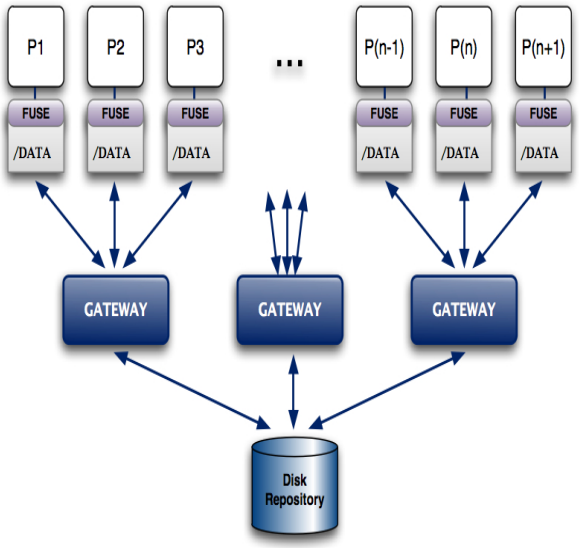

Figure2: Intermediate Gateway Service

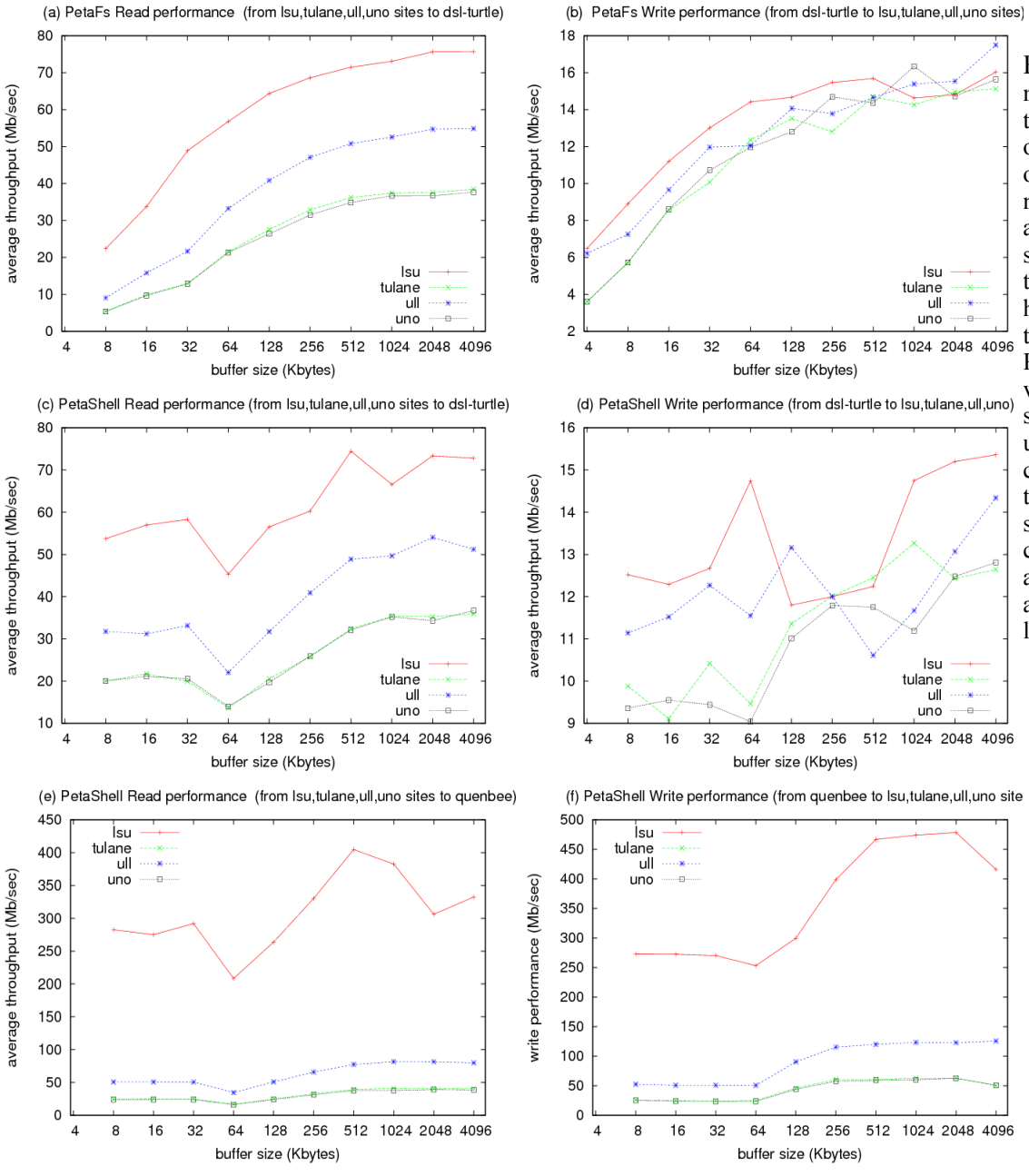

Figure3: There are 4 major remote sites and the metadata database is on *lsu* site. *Dsl-turtle* is outside of the LONI network and it has slow access to 4 PetaShare sites. *Queenbee* is inside the LONI network and it has much faster access to all of those 4 sites. Results are average values of 3 to 5 separate runs. We have used *cp* command and collected average throughput of 3 to 5 separate runs for copying 1MB, 10MB and 100MB files. The x-axis (buffer size) is in log scale.