International Conference on Information & Communication Technologies: From Theory to Applications, 19-23 Apr. 2004, Damascus, Syria

25851

25850

Data Transfer Management in Grid-based Mass Storage Environment

Putra Sumari, Fazilah Haron, Gian Chand Sodhy, Chan Huah Yong
School of Computer Sciences,
Universiti Sains Malaysia,
11800 Penang, Malaysia
{putras, fazilah, sodhy, hychan@cs.usm.my}

Abstract

The drastic increase in the data requirements of scientific applications and collaborative research has resulted of transferring a large amount of data among participating sites. The heterogeneous nature of mass storage systems employed by the different sites in Grid environment has made the data transfer among them a difficult problem. The general tendencies of data transfer are lacking of common interface between nodes, using a traditional simple scripts, dumping data to tapes and even using postal service. We introduce a framework of data management in transferring the data between heterogeneous mass storage in Grid environments. The system is capable of finding the suitable routes for data transfer between remote nodes without much user intervention.

1. Introduction

In recent years the amount of data for collaborative research such as in scientific applications has increased dramatically. They grow in petabyte and terabyte capacity. The amount of data that has to be transferred between remote participating sites is also increased. The success of transferring data at the right time and place is indeed important for the successful of computational job. As Grid evolves and data are stored at different sites, a smooth transferring mechanism is necessary. Mobile protocols such as GridFTP[1], Disk-router and even FTP have been developed to support the transfer of data between sites. However in many cases, due to the lack of a common interface and the know-how to perform high performance bulk data transfers, researchers have resorted of sending data either by using postal service in tapes form or simple script and manual. Mass-storage system protocols on the other hand are designed for local-area access and may not work well in the wide-area coverage. The resource storage manager (SRM), which acts as a middleware component that can interact with operating system and mass storage system to perform file archiving, file staging and file transferring are limited to a small coverage of network. The SRM has advantages in coordinating local storage resources such as local policy administration, shared resources monitoring and client request management should be applied to Grid-based mass storage environment. SRM normally is designed to handle local storage management and limited work has been done in interacting and exchanging information among remotes SRM. Indeed SRM plays an important role in Grid environment. There is no documented work on designing a schematic data transfer management for a Grid-based mass storage environment. In this work, we present a framework of data transfer between remote nodes in Grid environment. We introduce a SRM system that is embedded in Grid-based system which is capable to communicate with other remote nodes to find the suitable route for data transfer between them. Our system communicates with other nodes and able to make decision in finding the suitable cache nodes for routing data from source to destination. The system minimizes the problem of common interface compatibility amongst nodes during data transfer. Furthermore the transfer is carried out without much administrator intervention.

Section 2 outlines related studies. Section 3 presents our proposed data transfer management system. Section 4 discusses on experimental settings and expected results, and finally section 5 is the conclusion.

2. Related Study

In [6], tapes were used in sending visualization data via Federal Express from Los Alamos National Laboratory (LANL) to Sandia National Laboratory (SNL). It was faster than electronically transmitting them via TCP over the 155 Mbps(OC-3) WAN backbone. In [4], an ad hoc data pipeline was built between California and Illinois. The work was focused on handling error during transmission. While at Lawrence Berkeley National Laboratory, the management of tertiary storage for transferring mechanism of high energy physic analysis data within local network was established [5].

GridFTP introduced by Allcock[1] is capable of transferring data efficiently and secure in high bandwidth wide area network. Later, a Reliable File Transfer Service(RFT) [11] was introduced on top of GridFTP which allows byte streams to be transferred in

0-7803-8482-2/04/$20.00 ©2004 IEEE.

a reliable manner. RFT can handle a wide variety of problems such as dropped connections, machine reboots, and temporary network outages automatically via retrying. GridFTP and RFT are example tools that can be used to move data between systems which support their interface, but they can not be used to move data between heterogeneous storage systems with no (or minimal) common interface.

Koranda introduced Lightweight Data Replicator (LDR) [9] that can replicate data sets to the member sites of a DataGrid. LDR was combined with Globus [8] and later used for replicating LIGO [10] data. Its goal is to use the minimum collection of components necessary for fast and secure replication of data. Both, RFT and LDR used GridFTP to transfer data. These systems were carried out on the systems, which do not support a common data transport protocol. Our work focuses on system that lack of common interface between nodes. In [13] used Storage Resource Managers (SRMs) on top of their storage system to provide unified interface environment. Currently, a few data storage systems works have been established -- HPSS [12], Jasmin [3] and Enstore [7] – all support SRMs. The SDSC Storage Resource Broker (SRB) [2] aims to provide a uniform interface for connecting to heterogeneous data resources and accessing replicated data sets. SRB uses a Metadata Catalog (MCAT) to provide a way to access data sets and resources based on their attributes rather than their names or physical locations.

3. Data Transfer Management System

Our data transfer management system consists of four components: a Request Interpreter, a Transfer Planner, a Site Profiler and a Storage Resource Manager (SRM). The architecture of the system is shown in figure 1.

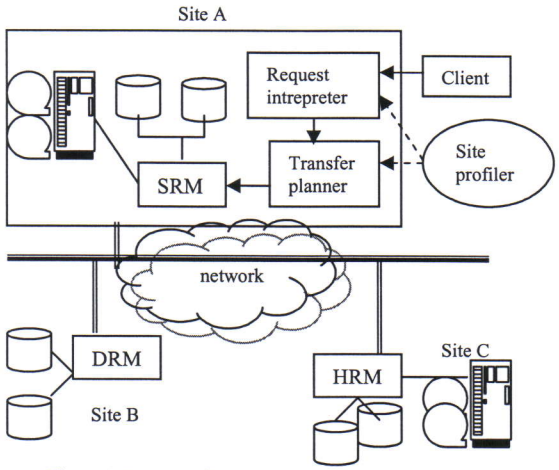

Figure 1: Data transfer management system architecture

Request interpreter accepts requests and interpret them to obtain query requirement such as file location, file size, destination address and etc. Transfer planner is the most important component in the system. It determines the route taken to deliver data from source to destination. The transfer planner makes decision on what routes should be taken with the help of site profiler. Site profiler is the component that will communicate with remote sites to obtain information such as the site location, disk space and transfer protocols. SRM is responsible to fetch data from storage area and transfer to the network according to the plan produced by transfer planner. A simple scenario would be a client wants to transfer file from site A to site C (see figure 1). Request interpreter accepts query and interprets to obtain file size, file location and destination address. The transfer planner then found that site A and C do not have a common interface. The transfer planner then scans of all neighboring nodes of site C looking for suitable cache nodes candidate. Transfer planner picks site B as a disk cache. The SRM then fetches the data from local disk and sends to B and then from site B to site C. SRM will communicate to all remote nodes in coordinating the task. The following sections describe each component in detail.

3.1 Request interpreter

The request interpreter provides a graphical user interface with a query language format to clients for inserting the requests. It then interprets the query to obtain file name, file size, file location (disk, optical, tape), and destination address. These data are later passed to transfer planner.

3.2 Site Profiler

Site profiler communicates to the destination site. It sends message to destination site to obtain the following information (we referred as basic information): the common interface, transfer protocol, size bandwidth between the source site and the destination site, space available at destination site. Apart from these, the site profiler will also scan all neighboring nodes to obtain basic information. This information is stored in profiler database.

3.3 Transfer planner

Transfer planner is the most important component in our data transfer management system. It determines the route that should be taken to deliver the data from source to destination. At present our system provides three routes of data pipeline between sites. The next section elaborates in detail how transfer planner makes decision.

3.3.1 Route 1

Let us assume that site A wants to send data to site C. The transfer planner communicates with site profiler to check the common interface compatibility, data transfer protocol and space availability at destination site. If there exist a direct interface between the two and there is available space at site C, the task is then passes to SRM. SRM will communicate with site C for delivery arrangement and use the underlying protocol of both sites to send data to site C. This is shown in figure 2.

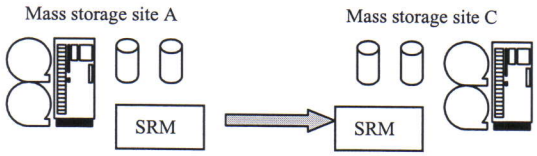

Figure 2: Direct route between two mass storage sites

3.3.2 Route 2

The second scenario would be site A wanting to send data to site C. Transfer planner communicates with site profiler to check the common interface compatibility, data transfer protocol and space availability at both sites. Assuming that there is no direct interface between them, the transfer planner will search in the profiler database for available neighboring nodes for a disk cache. In this case site B is the only candidate. Site B is chosen simply due to the common interface compatibility between both sites, A and C. Furthermore there is available space at site B which can act as a cache. Then SRM is instructed to handle the transfer from site A to site B and finally from site B to C. The transferring from site A to site B uses the underlying protocol of site A. The disk cache from site B to site C uses the underlying protocol of site C. This is shown in figure 3. If site B has limited storage space as a cache medium we have to remove the files to create space for the next transfer. The staging of transferring is built using DAG process as shown by figure 4.

The disk cache can be selected either a neighbor of site A or site C. This is shown in dash arrows as in figure 3. The figure shows the selected disk cache which is nearest to site A. The transfer mechanism is similar as outline for route that site B is selected as disk cache.

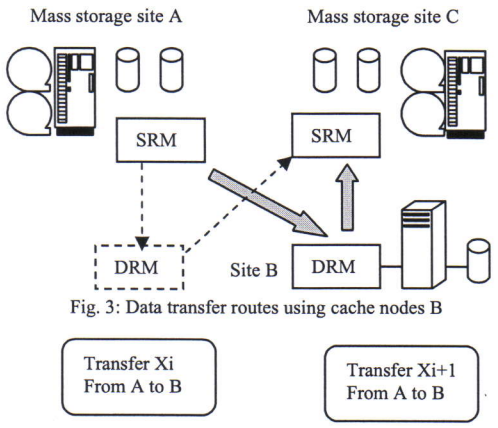

Fig. 3: Data transfer routes using cache nodes B

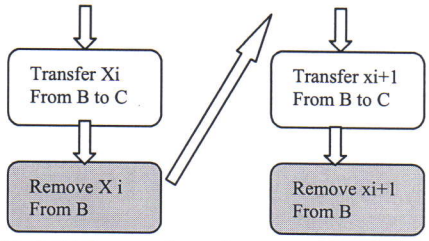

Fig 4: DAG process for staging file among site B, site B and site C.

3.3.3 Route 3

The third scenario is site A wish to send data to site D. The transfer planner found that there is no common interface between them. It then chose site B and site C as disk caches nodes. Since site B (disk cache) is nearest to site A and site C (disk cache) is nearest to site D, SRM transfers the data first from site A to site B using underlying protocol of site A, and then from site B to site C using the underlying protocol of site C and finally site C to site D using the underlying protocol of site D as shown in figure 5. Due to the limited space at site B and C, the SRM of both sites have to make sure to remove the file whenever done to create space for the next file. Figure 6 shows the DAG process of staging file among sites, site A, B, C and D.

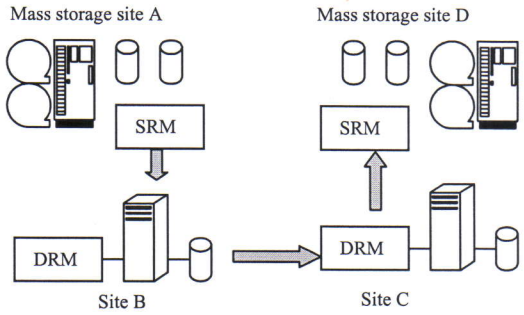

Fig 5 : Data transfer routes using cache node B and C

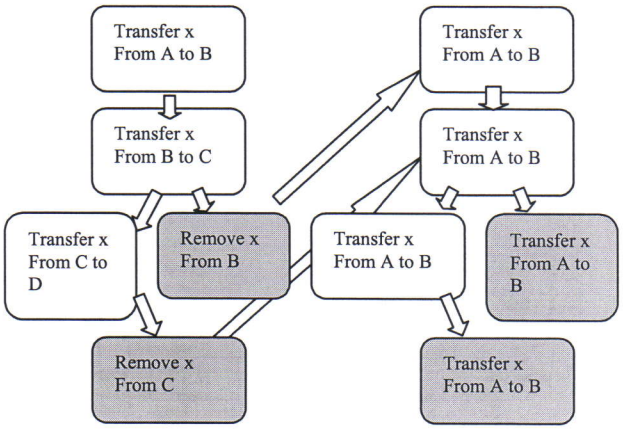

Fig 6: DAG process of staging file among four sites: site A, site B, site C and site D

3.4 SRM

Storage resource manager interacts with operating system and mass storage system, to perform file transfer, file archive and invoke middleware components (such as GridFTP) to perform file transfer operations. SRM consist of disk storage manager (DRM), tape resource manager (TRM), and hierarchical storage manager (HRM). The SRM also interacts to other sites to coordinate the data delivery. We applied the existing method in the literature available to perform the task.

4. Experimental settings and expected results

In our experiment, we will test our system between two remote sites: USMnet1 mass storage server in Penang and USMnet2 mass storage server in Kuala Lumpur. Both sites are 200 Km in distance. We also have set up several caches nearest to both sites. The total of data to be transferred from USMnet1 to USMnet2 is around 3 TB (3000 files of 1.0 GB each). There is no direct interface between USMnet1 and USMnet2. Space available at caches nodes during experiment is set to 10 GB. All servers have 100 Mb/s fast Ethernet card installed.

4.1 Performance of different routes

For route 2 delivery: data transfer used the underlying protocol of USMnet1 to cache node, then from cache node to USMnet2 using the underlying protocol of USMnet2. For route 3 delivery: data transfer used the underlying protocol of USMnet1 to cache node of USMnet1, then from cache node of USMnet1 to cache node of USMnet2 using underlying protocol of cache node of USMnet1, and finally from cache node of USMnet2 to USMnet2 using the underlying protocol of USMnet2. For both routes we are having cache nodes at both sides and that therefore we had control over the cache nodes in the case of we had no control of both USMnet1 and USMnet2. It was easy to deal with interface compatibility by having flexibility on cache nodes. For all routes average of 10 image files transmission at one time using GridFTP with DAG process applied at neighbor servers. The reason is simply due to the limited space at cache node which is only 10 MB, the end to end transfer is around 46 Mb/s. When we used disk router instead of GridFTP of route 3 and we found that we have smooth data transfer with disk router compare to using GridFTP. This is because the auto tuning of mechanism of disk router has automatically optimized the workload. This has shown that adding cache nodes in between do not affect the overall system.

The most significant advantage of having cache nodes at both sides was that we had control over those cache nodes, whereas we had no control over the USMnet1 and the USMnet2 mass-storage servers. To transfer data to/from the mass-storage servers, we need to use whatever protocol they provide. These protocols may not work well under certain conditions or may have certain limitations and may not allow us to tune their performance. This is a drawback of for wide-area transfers as they benefit considerably from tuning. With a source and a destination cache node, we made the mass-storage system work in the environment of local-area (where they are known to work well) and we have ability to choose an appropriate protocol for the wide-area transfer and got the ability to perform the necessary tune-up.

5. Conclusion

In this paper, we have shown a mechanism of data transfer management in mass storage system within wide area network. Our method allows a smooth data transfer between heterogeneous systems without a common interface. The system automatically finds suitable cache nodes to route data delivery. We present our method as an alternative of using tapes by postal service and writing scripts for handling data transfer and failures. We will perform an experiment to show that by adding additional nodes the end-to-end performance does not necessarily decrease but may in fact improve performance if done properly. We plan to look into handling failures in automated manner and automatic tuning of our system during the data transfer as well as incorporating searching of optimal routes.

6. References

[1] B. Allcock, J. Bester, J. Bresnahan, A. Chervenak, I. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnel, and S. Tuecke. Secure, efficient data transport and replica management for high-performance data-intensive computing. In *IEEE Mass Storage Conference*, San Diego, California, April 2001.

[2] C. Baru, R.Moore, A. Rafasekar, and M. Wan. The SDSC Storage Resource Broker. *In Proceeding of CASCON* Toronto, Canada, 1998

[3] I. Bird, B. Hess, and A. Kowalski. Building the mass storage system at Jefferson Lab. *In Proceeding of 18th IEEE Symposium on Mass Storage Systems*, San Diego, California, April 2001.

[4] T. Kosar, G. Kola, M. Livny. Building Data Pipe-line for High Performance Bulk Data Transfer in a Heterogeneous Grid Environment. Technical report University of Wisconsin, Computer Sciences Department, 2003.

[5] L. M. Bernardo, A. Shoshani, A. Sim, H. Nordberg, Access Coordination of Tertiary Storage for High Energy Physics Applications. Technical report of Lawrence Berkeley National Laboratory Berkeley 2000.

[6] D. Koester. Demonstrating the TeraGrid - A Distributed Supercomputer Machine Room. *The Edge, The MITRE Advanced Technology Newsletter*, 6(2), 2002.

[7] FNAL. Enstore mass storage systems. http://www.fnal.gov/docs/products/enstore/

[8] I. Foster and C. Kesselmann. Globus: A Toolkit-Based Grid Architecture. In *The Grid: Blueprints for a New Computing Infrastructure*, pages 259–278, Morgan Kaufmann, 1999.

[9] S. Koranda and B. Moe. Lightweight Data Replicator. http://www.lsc-group.phys.uwm.edu/lscdatagrid/LDR/overview.html, 2003.

[10] LIGO. Laser Interferometer Gravitational Wave Observation. http://www.ligo.caltech.edu/, 2003

[11] S. Son. Network Bandwidth Regulation In Cedar, 2003.

[12] SDSC. High Performance Storage System(HPSS). http://www.sdsc.edu/hpss/.

[13] A. Shishani, A.Sim, and J. Gu, Storage Resource Managers: Middleware Components for Grid Storage. *In nineteenth IEEE Symposium on Mass Storage System*, 2002

[14] D. Thain, and M. Livny. The Ethernet approach to grid computing. *In Proceeding of the twelfth IEEE Symposium on High Performance Distributed Computing*, Seattle, Washington, June 2003

[15] D. Thain, J. Basney, S. Son, and M. Livny. The kangaroo approach to data movement on the grid. In *Proceedings of the Tenth IEEE Symposium on High Performance Distributed Computing*, San Francisco, California, August 2001.