IJCSC International Journal of Computer Science & Communication Vol. 1, No. 1, January-June 2010, pp. 29-32

Parallel FTP: Clusters using Mirror Servers in Grid

N. K. Prema[1], K. Prasadh[2] & A. Arul Lawrence Selvakumar[3]

[1]Research Scholar, Department of CSE, VMKV Engineering College, Salem, India

[2]Professor & Head, Department of CSE, VMKV Engineering College, Salem, India

[3]Professor & Head, Department of CSE, MITS, Madanapalle, A.P, India.

[1]Premasenthi@gmail.com, [2]kprasadh@yahoo.com, [3]aarul72@hotmail.com

ABSTRACT

Grid computing is the widely used resource sharing mechanism in the large-scale distributed environment. Data grids are one of the storage resources sharing mechanism in the grid environment. Mirror servers are used for the fault tolerance requirements. Parallel File Transfer Protocol (PFTP) introduces the concept of data transfer using multiple parallel data paths between clusters and mirror servers. The new proposed system "Parallel file Transfer system for the Grid Environment" makes use of striped data distributed across the multiple IO-nodes of a Parallel File Access on clusters and mirror servers. This system is implemented as three applications such as grid server, mirror server and grid client. The grid server and mirror server applications are designed to handle the data updating and sharing process. The grid client is designed to carry out the file upload and download activities. The grid client can concurrently update the same file content into the grid server and set of mirror server while using the parallel input/output mechanism and multiple TCP connections. The protocol uses the striping format of the Parallel File Access to push multiple stripes of the file over multiple TCP connections. The flexibility in PFTP allows user to use different file transfer applications over these TCP connections. This system is implemented using java language to support cross platform environment.

Keywords: Parallel File Transfer Protocol, Transmission Control Protocol, Network File System, Network File System, Distributed File System.

1. INTRODUCTION

1.1. Grid Computing

A grid is a collection of geographically dispersed computing resources, providing a large virtual computing system to users [1]. The objective of a data grid system is two-fold. First, to integrate heterogeneous data archives stored in a large number of geographically distributed sites into a single virtual data management system. Second, to provide diverse services to fit the needs of high-performance distributed and data-intensive computing [2]. There are four commonly used kinds of resources in grids: computation, storage, communications, and software. Storage is viewed as the second most commonly used resource in a grid. The grid that presents an integrated view of storage is coined as "Data Grid". Memory, hard disk, and other permanent storage media are referred to as storage resources. This system particularly interested in secondary storage systems in grids, since secondary storage is 1000 times slower than main memory attached to a processor [2]. Many Networked file systems, which exhibit security and reliability features, have been widely applied to data grid. These file systems include: Network File System (NFS) [5], Distributed File System (DFS), Andrew File System (AFS), General Parallel File System (GPFS), and Parallel Virtual File System (PVFS). The amount of scientific data generated by simulations or collected from large-scale experiments is generally large, and such data tends to be geographically stored across wide-area networks for the sake of large-scale collaborations.

The notion of computational grid has been proposed for several years, mainly focusing on effective usage of global computational and network resources in a wide area network/Internet environment. However, the performance of a computational grid, where the effective usage of storage resources is ignored, will be degraded substantially if vast majority of applications running in the grid are data-intensive. To overcome this problem, various techniques have been developed and incorporated into a so-called data grid infrastructure.

Multiple TCP connections can be established in parallel GridFTP using OPTS RETR command. With this mechanism, a single file can be transferred from a single server through multiple TCP connections. Higher throughput can be expected by aggregating multiple TCP connections in comparison to using a single TCP connection. This can be explained by the following reasons: (1) by aggregating multiple TCP connections, larger bandwidth can be gained in the TCP congestion avoidance phase than those gained by other competing

TCP connections. This situation results from the fact that, in the TCP congestion avoidance phase, the AIMD window flow control is being executed, and therefore aggregated multiple TCP connections can transfer data more advantageously through the network with smaller packet loss probability. (2) by aggregating multiple TCP connections, the total TCP socket buffer size available to the file transfer increases [10]. Aggregation of N TCP connections can utilize as N times TCP socket buffer size as that of a single TCP connection. (3) by aggregating multiple TCP connections, ramp-up time of the transfer rate in the TCP slow-start phase is shortened.

2. RELATED STUDY

Multiple TCP connections can be established in parallel GridFTP using OPTS RETR command. With this mechanism, a single file can be transferred from a single server through multiple TCP connections. Higher throughput can be expected by aggregating multiple TCP connections in comparison to using a single TCP connection. This can be explained by the following reasons: (1) by aggregating multiple TCP connections, larger bandwidth can be gained in the TCP congestion avoidance phase than those gained by other competing TCP connections. This situation results from the fact that, in the TCP congestion avoidance phase, the AIMD window flow control is being executed, and therefore aggregated multiple TCP connections can transfer data more advantageously through the network with smaller packet loss probability. (2) by aggregating multiple TCP connections, the total TCP socket buffer size available to the file transfer increases [19]. Aggregation of N TCP connections can utilize as N times TCP socket buffer size as that of a single TCP connection. (3) by aggregating multiple TCP connections, ramp-up time of the transfer rate in the TCP slow-start phase is shortened. In the slow-start phase, the congestion window doubles every round-trip time. For this reason, through the aggregation of N TCP connections the speed of the transfer rate increase becomes N times as fast as that of a TCP connection. However, the throughput drops if the number of aggregate TCP connections, N, becomes too large, since this may cause the following situations: (1) the window size per TCP connection becomes smaller, so TCP timeout would frequently occur. (2) The overhead required for the server to process a TCP protocol stack would increase. Therefore, the optimal number of TCP connections, N, should be determined based on the network status. Nevertheless, how to optimize the number of parallel TCP connections, N, in GridFTP has not sufficiently been studied and still remains as an open issue.

3. SYSTEM DESIGN

3.1. Parallel File Transfer

Multiple TCP connections can be established in parallel GridFTP using OPTS RETR command With this mechanism, a single file can be transferred from a single server through multiple TCP connections. Higher throughput can be expected by aggregating multiple TCP connections in comparison to using a single TCP connection. This can be explained by the following reasons: (1) by aggregating multiple TCP connections, larger bandwidth can be gained in the TCP congestion avoidance phase than those gained by other competing TCP connections. This situation results from the fact that, in the TCP congestion avoidance phase, the AIMD window flow control is being executed, and therefore aggregated multiple TCP connections can transfer data more advantageously through the network with smaller packet loss probability. (2) by aggregating multiple TCP connections, the total TCP socket buffer size available to the file transfer increases[10]. Aggregation of N TCP connections can utilize as N times TCP socket buffer size as that of a single TCP connection. (3) by aggregating multiple TCP connections, ramp-up time of the transfer rate in the TCP slow-start phase is shortened. In the slow-start phase, the congestion window doubles every round-trip time. For this reason, through the aggregation of N TCP connections the speed of the transfer rate increase becomes N times as fast as that of a TCP connection. However, the throughput drops if the number of aggregate TCP connections, N, becomes too large, since this may cause the following situations: (1) the window size per TCP connection becomes smaller, so TCP timeout would frequently occur. (2) The overhead required for the server to process a TCP protocol stack would increase. Therefore, the optimal number of TCP connections, N, should be determined based on the network status. Nevertheless, how to optimize the number of parallel TCP connections, N, in GridFTP has not sufficiently been studied and still remains as an open issue.

3.2. Implementation

The grid environment can be applied to various types of application. Data grid and computational grid are the two widely used applications. The data grid serves as data distribution as manager among the grid clients. Large amount of data files can be transferred through the grid clients. The grid server also maintains the collection of large sized data files. These files can be accessed by remote clients. The clients can also upload their data files for the shared pool. Uploading and downloading activities are manages and monitored by grid server.

This system is simulated as client/server application under the personal computer based network environment. These types of network environments for the purpose of grid computing tasks are referred as "PC Grid". This system is divided into three major applications. They are the grid server, mirror server and grid client application. The grid server takes the role of controlling authority for the entire grid environment. The mirror server is the similar application like the grid server. This application act as backbone for the grid server application. The grid client application is designed as client for the grid environment.

3.3. Grid Server

The grid server is the centralized authority for the grid environment. This grid server application is designed to handle the data grid functionalities. User authentication, storage management and user management are the essential activities of the data grid environment. This system is divided into two major sub modules. They are user management and storage management. The user management sub modules deals with the transaction for new and registered users. The storage management carries out the data upload and downloads activities [3]. The user management sub module designed to handles the new user registration, user authentication and user control activities [2]. The new user registration process is done at the server side. The grid server administrator creates all the user accounts only. The user authentication operation is done at the time of log in request received from the grid clients. The administrator also removes the user accounts from the user list. The system supports the three types of access privileges. They are read, write, Read/write. All the uploading and downloading activities are initiated with reference to these access privileges. All the data storage and transmission operations are handled by the storage management. The upload and download activities are started after getting user authentication from the user management module. The storage management displays the list of shared files with their details. The grid client can download files from the shared file list. The authorized clients can also upload files to the shared data pool. The data receiving process can be done in different set of data paths and different sized segments.

3.4. Mirror Server

The mirror server application is similar to the grid server application. But it doesn't project as the entryway for the grid server clients. The mirror server takes the role of grid server [7] when the grid server is failed. A grid environment can have one or more mirror servers to assure the data transmission and data uploading activities. In some case the mirror server also takes the role of load balancing systems [8]. The mirror server also maintains the similar contents to the grid server. The mirror server contents are updated whenever the grid server contents are updated. They are data updating from the grid server and data updating by the grid clients [8]. This system uses the direct data updating from the clients. In this system the direct data updating from the client mechanism is selected. This mechanism supports the concurrent data updating for the server and also the mirror server. The mirror server updating can be done in any number of mirror servers at a time.

4. GRID CLIENT

The grid client application is designed as client application for the data grid environment. The data gird environment supports only for the data sharing activities. In this environment users can download and upload files with reference to their access privileges. The client application started with the user authentication process [6]. The log in form is the initial entry form for the application. The user should provide the user name, password and server name or address. The authentication task is carried out with this information. In the case of server failure, the user should enter the mirror server name. The grid client application is divided into three major modules. They are file upload, file download and user properties. The file upload module is designed to upload the files from the clients to the grid server. The download module is designed to download the shared file from the grid Sevres. The user property module shows the grid client information.

5. FILE UPLOAD PROCESS

The file uploading activities can be done with different criteria. This system supports the parallel data path based data transmission techniques. Generally file transfer protocol is designed with the single data path mechanism [7].The parallel file transfer protocol supports the multi path data transmission mechanism. In this mechanism user can choose the number of data paths of the uploading process. The grid server also tuned into requested data paths. This system also supports the dynamic sized segments. The File uploading process initially collects the file name, file description, size, data path count and the size of single segments. The server is initiated the setup the data path for the file transmission process. The client also connects with the mirror server and carryout the same process. The file is stripped with reference to the segment size. The stripped segments transferred concurrently to the mirror server and also the grid server. The mirror server count may be one or more. The transmission report shows the transactional details for the file uploading process.

6. SIMULATION RESULTS

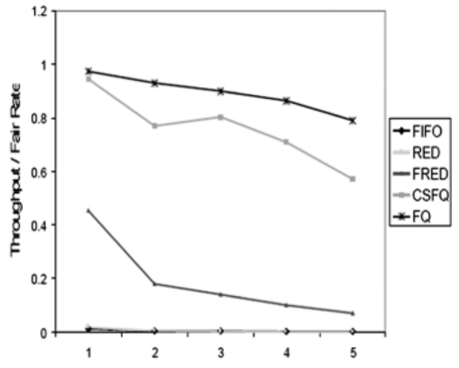

Fig 1: Relative Throughput of a TCP Over Multiple Grid Systems.

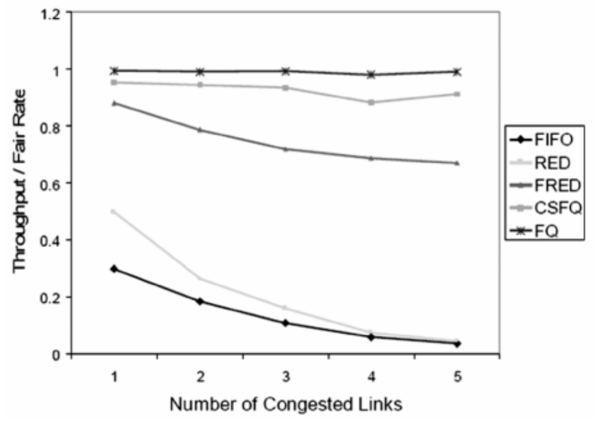

Fig 2: Relative Throughput of Multiple Clients Varying According Throughput based on Previous Algorithm.

7. CONCLUSION

The Parallel File Transfer System for the Grid Environment is developed to handle the file upload and download activities for the multi server grid environment. The parallel file transfer protocol (PFTP) can be used to carry out the parallel data transfer tasks among the clustered servers. This system is implemented to achieve the parallel file transfer achievement among the mirror servers. The individual data transmission for a server also done with the multiple parallel data paths. Grid server, mirror server and grid client applications are implemented separately. But the application test environment is build with one grid server, a set of mirror servers and a set of grid clients. File upload and download activities are tested with different number of mirror server client count. File upload process is intensively monitored for its performance. File upload process is for the same test environment is done with the different data paths and dynamic data segments. The performance of the file uploading process is better than the existing file transmission methods. Parallel data paths within a server and parallel data paths between the servers are the key features of the system [9]. File stripping mechanism added more flexibility for the data segment conversion process. The system is implemented using the java language.

REFERENCE

[1] D. Bhardwaj, "Application I/O on a Parallel File System for Linux Clusters", Chapter 25 in "High performance computing: Paradigm and Infrastructure", John Wiley & Sons, Inc, 2005.

[2] Foster, I., Kesselmann,C.(Editors): The Grid2: Blueprint for a New Computing Infrastructure. Morgan Kaufmann, 2003.

[3] The Globus Project. http://www.globus.org.

[4] Particle Physics Data Grid, http://www.ppdg.net/

[5] The TeraGrid Project, http://www.teragrid.org Proceedings of the First International Conference on e-Science and Grid Computing.

[6] The Reality Grid Project, http://www.realitygrid.org

[7] International Virtual Data Grid Project, http://www.ivdgl.org/

[8] May John, M., Parallel I/O for High Performance Computing, Morgan Kaufmann Publishers, 2001.

[9] http://ais.cern.ch/projs/ccs/welome.html

[10] Philip H. Carns, Walter B. Ligon III, Robert B.Ross, and Rajeev Thakur, 2000, PVFS: A parallel file system for Linux Clusters. Proc. 4th Annual Linux Showcase and Conference, 317-327.