Journal of Network and Computer Applications 36 (2013) 1116–1130

Contents lists available at SciVerse ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

A dual-direction technique for fast file downloads with dynamic load balancing in the Cloud ☆

Nader Mohamed [a], Jameela Al-Jaroodi [a,*], Abdulla Eid [b]

[a] College of Information Technology, UAEU, PO Box 17551, Al Ain, United Arab Emirates
[b] Department of Mathematics, University of Illinois at Urbana Champaign, USA

ARTICLE INFO

Article history:
Received 4 March 2012
Received in revised form
15 December 2012
Accepted 13 January 2013
Available online 9 February 2013

Keywords:
Cloud computing
Replicated data servers
Distributed FTP
Heterogeneous systems
Load balancing

ABSTRACT

File downloads make up a large percentage of the Internet traffic to satisfy various clients using distributed environments for their Cloud, Grid and Internet applications. In particular, the Cloud has become a popular data storage provider and users (individuals and corporates) are relying heavily on it to keep their data. Furthermore, most cloud data servers replicate their data storage infrastructures and servers at various sites to meet the overall high demands of their clients and increase availability. However, most of them do not use that replication to enhance the download performance per client. To make use of this redundancy and to enhance the download speed, we introduce a fast and efficient concurrent technique for downloading large files from replicated Cloud data servers and traditional FTP servers as well. The technique, DDFTP utilizes the availability of replicated files on distributed servers to enhance file download times through concurrent downloads of file blocks from opposite directions in the files. DDFTP does not require coordination between the servers and relies on the in-order and reliability features of TCP to provide fast file downloads. In addition, DDFTP offers efficient load balancing among multiple heterogeneous data servers with minimal overhead. As a result, we can maximize network utilization while maintaining efficient load balancing on dynamic environments where resources, current loads and operational properties vary dynamically. We implemented and evaluated DDFTP and experimentally demonstrated considerable performance gains for file downloads compared to other concurrent/parallel file/data download models.

1. Introduction

Users rely heavily on the Internet to get the files and information they need and several organizations offer file/data download services and sites where users can get what they need. Examples of such files/data a user might need include installing new applications such as video games and office applications, upgrading and maintaining existing applications such as operating systems and virus protection software, and retrieving important information such as financial and geographic information. In addition users may also need to store and retrieve their own data on remote servers for safe keeping and/or to avoid spending on their own storage facilities. Due to the heavy loads several providers rely on replicated servers and storage infrastructures to supply users with the required files efficiently and increase availability. For example, replicated FTP servers are available over Grid environments on which different Grid applications and users can download and utilize them. These files are usually very large and contain important scientific experiments and observations results such as climate simulation data (Allcock et al., 2001b) and high-energy physics (Bunn and Newman, 2003). Yet many of these sites rely on the traditional file transfer protocols such as FTP implementations where a single connection is created for a file download. However, some effort was done to introduce parallelism and concurrency to increase the performance. These techniques would naturally require additional coordination and control overhead that varies depending on the target environment and the design used. As a result, such solutions become suitable to specific infrastructures, but could fail miserably in others. In particular, the Cloud environment has characteristics distinct from most other distributed environments: First, the resources available are geographically distributed over huge areas. Second, the resources are heterogeneous, thus having various platform architectures, operating environments and devices; also, the resources are connected through shared dynamic and heterogeneous communication infrastructures; and the different components in the environments in some situations

☆ A preliminary version of this work was published in the proceedings of the 11th IEEE/ACM international symposium on cluster, cloud, and grid computing (CCGrid). Newport Beach, California, USA; May 2011.

* Corresponding author. Tel.: +971 5013 80642.
E-mail addresses: nader.m@uaeu.ac.ae (N. Mohamed), jaljaroodi@gmail.com, jaljaroodi@ieee.org (J. Al-Jaroodi), eid1@illinois.edu (A. Eid).

1084-8045/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.jnca.2013.01.006

are independent, thus having different access, management and accounting policies. Thus these characteristics make it hard to introduce techniques that require coordination and/or run time monitoring and control among the different resources and servers. For example, if several Cloud servers have replicas of the files, only one of them will be serving a client. Therefore, load balancing techniques rely on distributing the multiple clients' requests among the servers to reduce waiting time for each client; however, this does not improve the total download time per client since it has to be done through a single connection on a single server.

To enhance the download time, the most obvious approach is to divide the file and allow multiple servers to send different parts of the file to the client. In this case there are several issues to be considered: first is to determine how the file should be divided; second is to devise a suitable coordination model and determine how much overhead the coordination will impose; then to determine how will the client manage the different parts and put them together correctly. Several approaches were introduced and many offer good enhancements over the traditional single connection download; however, they also introduce high overhead to achieve good performance and load balancing. We offer a new technique that enhances overall download time, while inherently achieves load balancing without additional overhead on the distributed servers containing the replicas. In our technique we download file partitions from available replicas on the servers and we rely on the characteristics of TCP (Www link, 2012i) to help in the ordering and reconstruction of the file from the delivered blocks on the client side. We also perform the control and coordination on the client side, which relieves the servers from any coordination tasks and as a result, we can distribute the download among any set of distributed servers regardless of their locations or communications capabilities among each other. Consequently, this works very well in the Cloud as we cannot impose high overhead among the replicated data servers due to the high distribution and irregular operating conditions. The simplest way to describe our proposed technique is using a physical example. Consider a long sidewalk that needs to be built using bricks and we have two workers to do the job. The supervisor could ask them to start together from one end and move towards the other side. However, this means they will be interfering with each other all the time since one has to wait for the other to lay his brick before proceeding with his own. Another approach is to divide the sidewalk into two parts and one worker will start from one side moving towards the center, while the other will start from the center towards the end. Yet if one worker is slower than the other, then the total time to finish will depend on the slower worker. Using our technique, the workers are instructed to start from either end of the sidewalk and move towards the center. This way they both will work independently, thus there will be no contention; and each will work in his own pace until they meet somewhere along the way. Therefore, they will both finish at the same time, but each one may finish a different portion of the sidewalk depending on his speed. Similarly if we have more workers, for example six, we could divide the sidewalk into three sections and let each pair of workers work on one section from opposite directions. If one pair is faster than the others and they finish their section early the supervisor will ask them to help another pair by starting their work from the middle of the slow pair's section and so on. Eventually everyone will finish at around the same time, while not having to even talk to each other.

In this paper we first cover some basic concepts on file transfer and FTP and the related work in the field in Section 2. In that section we also discuss the different efforts towards enhancing FTP performance and the current work in parallel and concurrent FTP approaches. In Section 3 we describe DDFTP and describe how the technique works in the dual- and k-server cases in addition to introducing the heuristics used to enhance the performance and reduce the number of control messages needed for reassignments. In Section 4 we offer an experimental evaluation of DDFTP and compare it to other available models. The experiments include both the dual- and k-server cases. Finally we conclude the paper in Section 5.

2. Related work

Distributed computing has evolved dramatically over the past few decades and one of the driving forces behind this development is the need to store and exchange data. As computing applications advance, files and data sets grow bigger all the time. As a result, efficient data storage, management and transfer techniques are very important. Several techniques were introduced starting from FTP, moving over to various enhancements of FTP and many other different approaches for data and file transfers. The Cloud environment is currently the ultimate distributed environment and it has a growing base of users that require various services including data and file storage and download. Unlike other environments, the challenges and obstacles on performance enhancements on the Cloud are much more complicated and require more investigation and better designs to achieve that. As multiple sites exist and replicas are being made of data sets on various locations, the file/data download operations and management has not been improved much. Thus large data sets suffer major delays during the exchange. Some effort to help in this regard were made such as the proposal of MetaCDN (Broberg et al., 2009), which helps in terms of reducing the complexity of dealing with multiple storage providers on the Cloud and providing a unified namespace. Another example is BitDew (Fedak et al., 2009), which is designed to address the issue of large-scale data management in dynamic, heterogeneous, volatile and highly distributed environments such as the Cloud and the Grid. It is a programming environment that offers automatic and transparent data management. Several other research and industry approaches exist, yet most of them address the general issues of data transfer, management and availability, yet very few look into ways to enhance performance or offer efficient load balancing techniques to offer better download speeds and better resources utilization.

Earlier we used the dual direction technique to enhance large message transfers using multiple network interface cards (NICs) (Mohamed et al., 2006). Later we tried to offer a way to enhance overall data download times from FTP servers through parallelization across available mirrored sites, while maintaining efficient load balancing with minimum overhead (Al-Jaroodi and Mohamed, 2011). To do that we started with exploring models relying on TCP (Www link, 2012i) and (Parziale et al., 2006), which provides reliable ordered communication. As a result applications built on top of it, can make use of these features and ensure ordered and reliable packet delivery. In addition TCP has become the de facto standard for the Internet and most distributed systems. This made TCP a preferred protocol to implement FTP software. The Standard FTP implementations mainly use TCP as the underlying communication protocol due to its reliability features.

The original FTP as described in RFC 959 (Www link, 2012b) was designed to support file transfer in a distributed environment using the client/server model and serving a single connection at a time. Several enhancements and extensions to the original FTP were made and published in different RFCs such as RFC 1579 (Www link, 2012c), RFC 2640 (Www link, 2012g) and RFC 3659 (Www link, 2012a). FTP can operate in one of three modes