Future Generation Computer Systems 26 (2010) 1127–1140

Contents lists available at ScienceDirect

Future Generation Computer Systems

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

Implementation of a medical image file accessing system in co-allocation data grids☆

Chao-Tung Yang[a,*], Chiu-Hsiung Chen[b], Ming-Feng Yang[a]

[a] High-Performance Computing Laboratory, Department of Computer Science, Tunghai University, Taichung, 40704, Taiwan

[b] Tungs' Taichung MetroHarbor Hospital, No. 699, Sec. 1, Jhongci Rd., Wuci Township, Taichung County 435, Taiwan

ARTICLE INFO

Article history:
Received 9 November 2009
Received in revised form
26 May 2010
Accepted 28 May 2010
Available online 8 June 2010

Keywords:
Medical images
Data grid
Grid computing
Co-allocation
File transferring

ABSTRACT

There are two challenges of using the PACS (Picture Archiving and Communications System). First, PACS are limited to certain bandwidths and locations. Second, the high cost of maintaining Web PACS and the difficult management of Web PACS servers. Besides, the quality of transporting images and the bandwidth of accessing large files from different locations are difficult to guarantee. For instance, radiologists make use of PACS information system for achieving high-speed accessing medical images. Physicians, on the other hand, utilize web browsers to indirectly access the PACS information system via non-high-speed network. The insufficient bandwidth may cause bottleneck under a host of querying and accessing. As hospitals exchange large files such as medical images with each other via WANs, the bandwidth cannot support the huge amount of file transportation. In this paper, we propose a PACS based on data grids, and utilize MIFAS (Medical Image File Accessing System) to perform querying and retrieving medical images from the co-allocation data grid. MIFAS is also suitable for data grid environments with a server node and several client nodes. MIFAS can take advantage of the co-allocation modules to reduce the medical image transfer time. Also, we provide experiments to show the performance of MIFAS. Furthermore, in order to enhance the security, stability and reliability in the PACS, we also provide the user-friendly management interface.

1. Introduction

Nowadays, 2D, 3D, and 4D medical imaging devices are increasingly needed by hospitals. With the progress of medical photograph, the resolution of medical images is raising. Therefore, the scale of medical image files range from MB to GB. The size of high-resolution medical images, such as 64/128-slice CT scans, 3.0T MRI, and PET, often exceed one hundred MB or more. However, the speed of progress on many high-quality imaging devices and it related infrastructure are not match. Current Picture Archiving and Communication Systems (PACS) [1–4] are unable to provide efficient query response services. It is difficult to sustain huge queries and file retrievals under limited bandwidth. Therefore, the quality of communication in the Web PACS network would be restricted by bandwidth and conventional access strategies about exchanging and downloading a large amount of images. In order to enhance the quality of medical treatment, the medical imaging needs to associate with efficiency file transfer strategy to achieve high-speed accessing.

In this paper, we present a new strategy for processing medical image queries, which is based on the co-allocation [5–10] strategy for data grid environments. A data grid is a system composed of multiple servers that work together to manage information and related operations – such as computations – in a distributed environment. Our proposed system is called the Medical Image File Accessing System (MIFAS) for co-allocation data grids. To solve these problems, we propose the PACS based on the co-allocation data grid environment. MIFAS helps us to transfer huge medical images into the co-allocation data grid environment. We utilize the Globus Toolkit 4.0.7 [11–13] to establish the data grid environment for deploying co-allocation strategy and processing medical images. MIFAS helps users to quickly retrieve medical images from Medical Data Grid. The Cyber Agent Service and the Grid Service GUI desk application are implemented to assist in query and retrieve medical images. Also, MIFAS provides resume broken transfer to deal with the unstable circumstance of network. It not only enhances the overall quality of medical care system but also supports multiple replicas of medical images for failover recovery.

This paper presents a strategy to improve the security, stability and reliability of the PACS. Our strategies focus on integrating the service of processing medical images and stimulating PACS architecture into grid environments. The remainder of this paper is

☆ This work is supported in part by the National Science Council, Taiwan R.O.C., under grant nos. NSC 97-2622-E-029-003-CC2 and NSC 98-2622-E-029-001-CC2.

* Corresponding author. Tel.: +886 4 23590415; fax: +886 4 23591567.
E-mail addresses: ctyang@thu.edu.tw (C.-T. Yang), t2884@ms.sltung.com.tw (C.-H. Chen), orsonyang@gmail.com (M.-F. Yang).

0167-739X/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.future.2010.05.013

organized as follows. Background review and studies are presented in Section 2. The Cyber Agent Transformer is introduced in Section 3. Experimental results are presented in Section 4. Section 5 concludes this article.

2. Background

2.1. Data grids

Data grids enable the sharing, selection, and connection of a wide variety of geographically distributed computational and storage resources for solving large-scale data-intensive scientific applications (e.g., high energy physics, bioinformatics applications, and astrophysical virtual observatory) [14–16]. The term "Data grid" traditionally represents the network system with distributed storage resources, from archival systems to caches and databases, which are linked using a logical name space to create global, persistent identifiers, and provide uniform access mechanisms [17,1,18–20,10].

Distributed scientific and engineering applications could access huge amounts of data between storage systems; these file often generated by many geographically distributed applications and users for analysis and visualization. Data grids consist of scattered computing and storage resources located in different countries/regions yet accessible to users. Data grid also provides file replication, which means datasets could be replicated within grid environments for reliability and performance. With replication, clients could discover existing data replicas and create or register new replicas.

Replica selection is important to data-intensive applications, it can provide location transparency. When a user requests a data set, the system determines an appropriate way to deliver the replica to the user. Another issue concerning replica selection is the prediction of the transfer time. Therefore, it involves the inspection of many characteristics and is a complex piece of work.

In situations where replicas are to be selected based on access time, Grid information services can provide information about network performance and perhaps the ability to reserve network bandwidth, while the metadata repository can provide information about the size of the file. Based on this, the selector can rank all of the existing replicas to determine which one will yield the fastest data access time. Alternatively, the selector can consult the same information sources to determine whether there is a storage system that would result in better performance if a replica was created on it.

2.2. Co-allocation model

The proposed architecture [6] consists of three main components: an information service, a broker/co-allocator, and local storage systems. Fig. 1 shows the co-allocation of data grid, an extension of the basic template for resource management [21] provided by the Globus Toolkit. Applications specify the characteristics of desired data, and pass attribute descriptions to a broker. The broker searches for available resources, gets replica locations from the Information Service [22] and Replica Management Service [7] to retrieve lists of physical file locations.

We implemented the following eight co-allocation [23] schemes including Brute-Force (Brute), History-based (History), Conservative Load Balancing (Conservative), Aggressive Load Balancing (Aggressive) [5], Dynamic Co-allocation with Duplicate Assignments (DCDA), Recursively Adjusting Mechanism (RAM) [23], Dynamic Adjustment Strategy (DAS) [24], and Anticipative Recursively Adjusting Mechanism (ARAM) [25]. In [5], the author proposes the co-allocation architecture for co-allocating grid data transfers across multiple connections by exploiting the partial copy feature of GridFTP. It also provides Brute-Force, History-Base, and Dynamic Load Balancing for allocating data block.

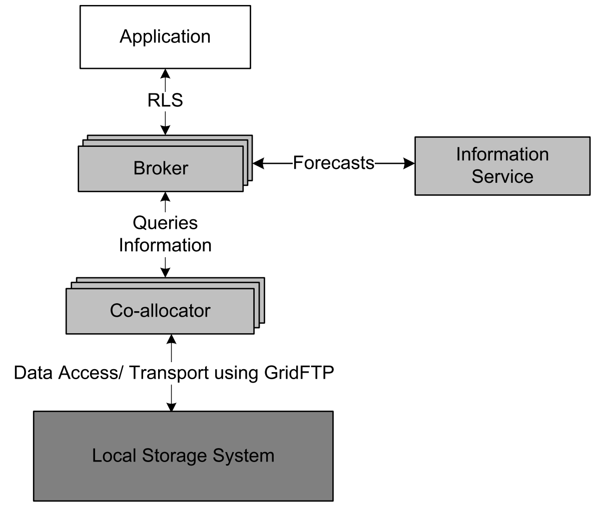

Fig. 1. Data grid co-allocation architecture.

- Brute-Force Co-Allocation: Brute-Force Co-Allocation works by dividing files equally among "n" available flows (locations). Thus, if the data to be fetched is size, "S" and there are "n" locations to fetch it from, then this technique assigns to each flow a data block of size, "S/n". For example, if there are three sources, the target file will be divided into three blocks equally. And each source provides one block for the client. With this technique, although all the available servers are utilized, bandwidth differences among the various client–server links are not exploited.
- History-based Co-Allocation: The History-based Co-Allocation scheme keeps block sizes per flow proportional to transfer rates predicted by the previous results of file transfer results. In history-based allocation scheme, the block size per flow is commensurate to its predicted transfer rate, decided based on a previous history of GridFTP transfers. Thus, the file-range distribution is based on the predicted merit of the flow. If these predictions are not accurate enough, renegotiations of flow sizes might be necessary as slower links can get assigned larger portions of data, which could be weight heavily on the eventual bandwidth achieved. With the history-based approach, client divides the file into "n" disjoint blocks, corresponding to "n" servers. Each server "i", $1 \le i \le n$, has a predicted transfer rate of "B_i" to the client. In theory then, the aggregate bandwidth "A" achievable by the client for the entire download is $A = \sum_{i=1}^{i=n} Bi$. For each server "i", $1 \le i \le n$, and for the data to be fetched is size of, "S", the block size per flow is $S_i = \frac{Bi}{A} \times S$.
- Conservative Load Balancing: One of their dynamic co-allocation is Conservative Load Balancing. The Conservative Load Balancing dynamic co-allocation strategy divides requested datasets into "k" disjoint blocks of equal size. Available servers are assigned single blocks to deliver in parallel. When a server finishes delivering a block, another is requested, and so on, till the entire file is downloaded. The loadings on the co-allocated flows are automatically adjusted because the faster servers will deliver more quickly providing larger portions of the file.
- Aggressive Load Balancing: Another dynamic co-allocation strategy, presented in [5], is the Aggressive Load Balancing. The Aggressive Load Balancing dynamic co-allocation strategy presented in [5] adds functions that change block size deliveries by: (1) progressively increasing the amounts of data requested from faster servers, and (2) reducing the amounts of data requested from slower servers or ceasing to request data from them altogether.
- Neither prediction nor heuristics approaches, the DCDA scheme dynamically co-allocates duplicate assignments and copes