A Parallel Method to Improve Medical Image Transmission

Rouzbeh Maani · Sergio Camorlinga · Neil Arnason

Published online: 12 May 2011
© Society for Imaging Informatics in Medicine 2011

Abstract The staggering number of images acquired by modern modalities requires new approaches for medical data transmission. There have been several attempts to improve data transmission time between medical imaging systems. These attempts were mostly based on compression. Although the compression methods can help in many cases, they are sometimes ineffectual in high-speed networks. This paper introduces parallelism to provide an effective method of medical data transmission over both local area network (LAN) and wide area network (WAN). It is based on the Digital Imaging and Communications in Medicine (DICOM) protocol and uses parallel TCP connections in storage services within the protocol. Using the proposed interface in our method, current medical imaging applications can take advantage of parallelism without any modification. Experimental results show a speedup of about 1.3 to 1.5 for CT images and relatively high speedup of about 2.2 to 3.5 times for magnetic resonance (MR) images over LAN. The transmission time is improved drastically over

WAN. The speedup is about 16.1 for CT images and about 5.6 to 11.5 for MR images.

Keywords DICOM · Telemedicine · Parallelism

Background

Medical data transmission is one of the most important areas in telemedicine. In the last decade, two phenomena have highlighted the key role of medical data transmission: medical data overload and emergence of real-time telemedical applications.

Data overload has become a challenge for current medical imaging systems in recent years. One reason is that the modern modalities can produce large number of high-quality images. For instance, a routine chest CT may produce 300 to 500 images and a CT angiography runoff study may include 1,500 to 2,000 images [1]. By considering the number of studies performed per patient and the number of patients examined each day, we can see the enormous load of data that medical imaging systems have to deal with. As a result, we need to find viable approaches for transmission, access, display, and navigation through these large amounts of data [2].

On the other hand, some of the modern medical imaging applications are real-time, interactive and collaborative, and as a result they need to transmit medical data as fast as possible. Some examples are presented in [3–6]. Prasad et al. [3] provided a tool for doctors to view medical images online and collaborate over the Internet. Marthinsen et al. [4] tried to integrate online PACS systems and multichannel video-conferencing to run multiple remote magnetic resonance (MR) laboratories. Wangeheim et al. [5] introduced a radiological collaborative tool that is based on the DICOM protocol and enables manipulating images as well as session recordings. Maani et al. [6] introduced a tele-

R. Maani (✉) · S. Camorlinga · N. Arnason
Department of Computer Science, E2-445 EITC,
University of Manitoba,
Winnipeg, MB R3T 2N2, Canada
e-mail: maani@cs.umanitoba.ca

R. Maani · S. Camorlinga · N. Arnason
TRLabs,
100-135 Innovation Drive,
Winnipeg, MB, Canada

S. Camorlinga
Radiology Department, Faculty of Medicine,
University of Manitoba,
Winnipeg, MB, Canada

R. Maani
Department of Computing Science,
University of Alberta,
Edmonton, AB T6G 2E8, Canada

medicine platform that supports a real-time, ubiquitous, collaborative, and interactive meeting environment equipped with 3D visualization facilities. Telemedicine applications and particularly teleradiology applications need to deal with the difficulty of transmitting large volumes of medical data subject to relatively low available bandwidth [7].

A common approach towards improving the speed of data transmission is to use compression techniques. There have been several attempts to improve the speed of data transmission between DICOM application entities. These attempts are mainly based on the compression idea [7–16]. However, on a reasonably fast network these techniques are not efficient because the overhead of compression and decompression slows down the total transmission operation instead of speeding it up [17]. In other words, compression techniques are ineffectual in high-speed networks.

Recently, Maani et al. [18] presented an approach that uses a combination of parallelism and compression to improve transmission time. They use a pair of interfaces to obviate any change in the current medical systems. However, the effect of pure parallelism (i.e., no compression) is not studied in their paper. In addition, they do not provide the architectural details of the interfaces. Finally, the dataset used in their experiments consisted of images acquired by different modalities and the effect of the modality type is unclear.

In this paper, we expand the research study presented by Maani et al. to assess the effect of pure parallelism. Moreover, we present the detailed architecture of the interfaces. Finally, the impact of image modality type is studied.

The proposed approach in this paper is based on the DICOM protocol [19]. DICOM is the world-wide de facto standard for interconnecting medical imaging systems because of advantages such as interoperability, integrity, and consistency [7, 20].

The proposed method uses parallel transmission control protocol (TCP) connections to carry out the image transformation between two DICOM application entities. These parallel connections are implemented in the DICOM protocol Storage Services. We use a pair of interfaces to provide parallelization. These interfaces have three main characteristics which make them interesting:

1. They parallelize image transmission automatically.
2. They use only one association (a DICOM term defined in the next section).
3. They do not impose any change to the current medical systems.

To provide parallelism, a multi-threading technique has been used in the interfaces. The implementation of the interfaces is in Java and is built on top of the dcm4che2 tool [21].

The rest of the paper is organized as follows: [DICOM Protocol](#) section provides an overview of the DICOM protocol and its terminology. [Using Parallel TCP Connections](#) section explains the proposed method. The experimental results are discussed in [Evaluation](#) section, and finally, [Discussion](#) section includes the conclusion and the future works.

DICOM Protocol

DICOM is the standard file format for medical images. It also includes a communication protocol for medical image transmission. This section provides a brief overview of the DICOM standard communication part [17] because the proposed approach is based on the DICOM protocol.

The DICOM protocol is based on the Open System Interconnection [22] network protocol's architecture, which consists of different layers. Each layer has some predefined tasks and the corresponding layers communicate to each other through predefined protocols. The most common architecture uses five layers. The layers from bottom to top are: physical, data link, network, transport, and application layers [23].

DICOM is an application-layer protocol (Fig. 1). It is compatible with the TCP/IP protocol suite and therefore can be used over the Internet. In DICOM terminology, an application using the DICOM protocol is called an application entity (AE).

Each AE can either request or provide one of the DICOM protocol services. The service is called service class. When an AE requests a service, it plays the role of a service class user (SCU) while the AE providing the service plays the role of a service class provider (SCP). For example, when a DICOM application wants to find an image in a DICOM server, the DICOM application is the SCU, the DICOM server is the SCP, and the DICOM FIND service is the service class (Fig. 2).

Each service class consists of data and a function related to that data. For example, an MR image can be bound with different functions such as printing or storing; therefore, in this case the MR image can be associated with different service classes.

Each Service Class consists of two parts:

1. The object instance (e.g., a CT image). This object is called the Information Object Definition (IOD).
2. The function or service for the object (e.g., storage) which is called the DICOM message service element (DIMSE).

Since each service class is a service paired with an object, it is called the service-object pair or SOP (Fig. 3).

Fig. 1 DICOM as an application layer protocol

When two AEs want to start communicating, they need to establish a session. This session is called association. Association establishment starts with exchanging some important information such as supported data encoding (i.e., transfer syntax) as well as services provided by the SCP. After this step, the SCU can request SOPs from the SCP. Finally, after the completion of SOP requests, the association is terminated (Fig. 4).

Using Parallel TCP Connections

This section explains the proposed method. In [The Scope](#) section, we discuss the DICOM protocol scope that we target in this paper. In [Parallel Data Transmission](#) section, we explain how we implement the parallel TCP connections. Finally, the detailed architecture of the interfaces is explained in [Interface Structure](#) section.

The Scope

An important DICOM protocol service is the storage SOP, which is responsible for transmitting images between two AEs. Since the storage SOP service carries the major data load, improvements in the storage SOP will influence the

Fig. 2 Service Class, Service Class User (SCU), and Service Class Provider (SCP)

total communication time between two AEs. A common approach is using compression techniques in the storage SOPs to reduce the data volume and consequently speeding up the transmission time. However, in many cases, especially in high-speed networks, the compression overhead degrades transmission time [17].

In our method, we use parallelism in the storage SOPs. The parallelism can also be combined with compression techniques. In this paper, we use pure parallelism in storage SOPs and study its performance and refer interested readers to [18] for a combined method study.

Parallel Data Transmission

To parallelize data transmission, we use a pair of interfaces, one for each AE. Each interface is located in the same location where the AE is located. These interfaces carry out the storage service in parallel; however, there is only a single connection between the AE and the corresponding interface. Figure 5 compares this method with current methods that use a single connection for storage. The single connection between the AE and the corresponding interface helps us to have only one association. In other words, from the AE's point of view there is only one connection (association), while automatically, the connection has been parallelized by the interfaces. As a result, without increasing the number of associations, we parallelize the connection.

Since each AE and its corresponding interface are located on the same computer, the data transmission

Fig. 3 Service-object pair structure

Fig. 4 DICOM communication steps

between them is carried out in real time. The connection between the two interfaces is in parallel. The parallelism makes better use of the available network bandwidth capacity and thereby speeds up data transmission.

By using the pair of interfaces, the need for changes in the current AEs is obviated. In other words, any two DICOM AEs can take advantage of using the parallel storage capability without change. Each AE sends and receives data to its corresponding interface. This communication is carried out by a single connection that uses the default Transfer Syntax. Moreover, by using a single connection for each AE, all data transmission is carried out in just one association. This feature is helpful if the number of associations is limited by the SCP.

Note that parallel connections are only established when an AE sends storage SOP requests to its corresponding

interface. The two interfaces use a single DICOM connection in other cases

Interface Structure

Unlike common AEs that have either the SCP or SCU role at a time, the interfaces have both roles and therefore they have a different structure. Depending on sending or receiving data, the interface plays one of the SCP or SCU roles. For example in Fig. 6, if we assume that AE 1 requests a CT image storage SOP from interface 1. AE 1 is the SCU and interface 1 is the SCP. Then this request is sent to interface 2. Now, interface 1 plays the SCU role while interface 2 plays the SCP role. Finally, the request is passed on to the AE 2 where interface 2 is the SCU and AE 2 is the SCP.

Fig. 5 Using a pair of interfaces (top) in comparison with a single connection (bottom)

Fig. 6 Assigning both SCP and SCU roles to the interfaces

We used the Java language and dcm4che2 tool [21] to implement the interfaces. Dcm4che2 is an open-source, java-based implementation of the DICOM protocol. We used a thread pool to make parallel TCP connections. The user can determine the size of this pool. Whenever an interface receives a message, the type of the sender is checked. If the interface is receiving a message from the other interface, the message is simply forwarded to the AE. If the interface is receiving a message from the corresponding AE, and the service is not a storage SOP, the message is forwarded using the single available connection; otherwise, a thread will be pulled from the thread pool and makes a new storage connection to the other interface (Fig. 7). We use no optimization method in the connections and the interfaces use common TCP connections. Each interface consists of five main components: receiver, sender to AE, sender to interface, message interpreter, and thread pool. The components of the interfaces and their links are depicted in Fig. 8. The control flow, single connection, and parallel connections are shown in different line styles.

Evaluation

To assess the efficiency of the proposed method, two factors were considered:

- The type of the modality by which the image was acquired: CT and MR images were tested in our evaluation.

- The underlying network type: local area network (LAN) and wide area network (WAN) were considered for the evaluation.

We used four datasets. Two CT and two MR image datasets were used for evaluation. By testing two different datasets for each modality, we could have more reliable results and therefore make better conclusions. Table 1 shows the dataset specifications. BREBIX [24] is a dataset including CT images of hypernephroma, arterial, and venous acquisitions. CARCINOMIX [24] includes lung carcinoma CT images. MR-BODY [25] collection includes MR images for different body parts including brain, abdomen, heart, knee, shoulder, and spine. MR-BRAIN [26] is a brain MR images dataset.

Evaluation Method

To evaluate the parallel method, in the first step, the datasets were sent by common transmission method (i.e., single connection) without using the pair of interfaces. Then we used the pair of interfaces and measured the transmission time. For both single and parallel methods, we used implicit VR little endian (i.e., the default transfer syntax in DICOM).

We used both LAN and WAN networks to study the efficiency of the method based on the network type. To evaluate the effect of parallelism, 2 to 30 parallel connections were used. The experiments were repeated 20 times on WAN and 70 times on LAN and the average speedup of the method was reported.

The receiver for both LAN and WAN experiments had a Core 2 Quad-Core CPU and 2 GB of RAM running Linux CentOS.

The LAN experiments were carried out on a 100 Mbps network. A Pentium IV computer with dual-core CPU and 1 GB of RAM running Windows XP is used as the sender.

Fig. 7 Flow diagram of message forwarding in the interfaces

Fig. 9 Speedup of the parallel method over LAN for the four datasets

sometimes remarkable. One of the most important factors is the network type. We obtained the best improvement in data transmissions over WAN.

We observed that the speedup range is different based on modality type. The transmission time improvement is better for MR datasets (i.e., MR-BODY and MR-BRAIN) over LAN. In addition the parallel method needs a small number of parallel connections to achieve the maximum speedup for MR images. While, the speedup for CT image datasets is slight over LAN and the number of parallel connections do not change the speedup noticeably.

The experiments also demonstrated that after using a certain number of parallel connections, the speedup does not increase by adding more parallel connections. This occurs because the parallel connection implementation imposes overheads (e.g., Java threads for our implementation) and because of network saturation. All network traffic is bound by the available bandwidth. As a result, increasing the number of parallel connections when the transmission

data is bound by the network's available capacity only adds overhead and decreases the speedup consequently.

The amount of speedup is remarkable over WAN. Unlike the LAN case, the parallel method had a better performance for CT datasets. The parallel method improved the transmission time for CT datasets about 16 times; however, it employed a relatively large number of parallel connections to achieve the maximum speedup.

Similar to the LAN experiments, the parallel method did not speed up transmission time after some levels of parallelism. The reasons are similar (i.e., overhead and network saturation).

If we compare the experimental results to the study performed by Maani et al. [18], we can see that we achieved similar results. However, they used a different dataset and combined the parallelism with compression. Their dataset includes 303 images (i.e., approximately 150 MB) consisting of CT, MR, digital X-ray, digital mammography X-ray, and X-ray radiofluoroscopic. They

Fig. 10 Speedup of the parallel method over WAN for the four datasets

reported speedups of transmission time of 1.22 times over LAN and 14 times over WAN.

In this study, we reached a higher speedup over LAN. Specifically, we observed that the speedup is improved remarkably for MR datasets. Over WAN, the speedup was higher for CT images and was lower for MR images in comparison with the results presented in [18].

By comparing pure parallelism and the combination of parallelism and compression, one might suggest that compression has a minor effect in the transmission speedup compared to parallelism. However, employing compression reduces network usage which is advantageous in all types of networks.

Future Works

This study shows similar results to [18]; however, we did not compare pure parallel method with the combined methods presented in [18]. One of the future works is comparing pure parallel method with the combination of parallelism and compression methods. This comparison can be extended to other modality-type datasets (e.g., US). Our proposed method can also be compared with asynchronous DICOM.

In this study, we used dcm4chee and Java for implementation. One possible future work is implementing the method using faster languages like C/C++. Other optimization methods (e.g., TCP/IP level optimizations) may also improve the results further.

We observed that there is an optimal number of parallel connections based on the network and image modality type which results in the maximum speedup. Finding that number and enhancing the interfaces to choose that number of parallel connections is also one of the future works.

Finally, the interfaces use TCP connections. There are other options such as secure FTP and web services that can be implemented in the interfaces. Then, we can study the role of the underlying protocol in the performance.

Conclusions

This paper presents a method based on the DICOM protocol to improve transmission time of medical data. We show that parallelism is an effective method to overcome this problem. The combination of compression methods and parallelism also improves transmission time over both LAN and WAN [18]. However, using pure parallelism provides similar results.

We presented the architecture of a pair of interfaces which automatically parallelize data transmission by using only one association. These interfaces obviate any change in the current medical systems to use the proposed parallel

method. Moreover, some medical applications do not allow more than one association at a time. The proposed method addresses this constraint and provides parallelism for them.

To examine the proposed method, we set up experiments over LAN and WAN using two datasets of CT and MR images. The experiments over LAN showed that the speedup was slight for CT image datasets (about 1.3 to 1.5) and was relatively high for MR image datasets (about 2.2 to 3.5 times). However, the parallel method improved transmission time drastically over WAN. The speedup was about 16.1 for CT image datasets and was about 5.6 to 11.5 for the two MR image datasets. The observations show that the modality type can play an important role in overall speedup.

Our observations corroborate the idea of Maani et al. [18] where using parallelism (with and without compression) speeds up transmission time over both LAN and WAN. They showed that compression methods (without parallelism) degrade transmission time over LAN whereas their method provided speedup over LAN. We observed that pure parallelism provides similar (and even sometimes better) speedup over LAN. We also observed that the speedup over WAN is remarkable; however, it depends highly on the modality type of the dataset.

References

1. Dreyer KJ: PACS: A guide to the digital revolution. Springer, New York, 2006
2. Andriole KP, Barish MA, Khorasani R: Advanced image processing in the clinical arena: Issues to consider. *Journal of the American College of Radiology: JACR* 3(4):296–298, 2006
3. Prasad D, Ray S, Majumdar A, Mukherjee J, Paul S, Verma A: Real time medical image consultation system through Internet. *Journal of Healthcare Engineering* 1(1):141–154, 2010
4. Marthinsen PB, Hald J, Bergstrom R, Jakobsen J, Christensen D, Roterud H, Yoo M, Larum F: Regional pacs and telemedicine. *International Journal of Computer Assisted Radiology and Surgery* 3(0):155–163, 2008
5. von Wangenheim A, Prusse M, Maia RS, Abdala DD, Regert AG, de Souza Nobre LF, Comunello E: Recording and reenactment of collaborative diagnosis sessions using DICOM. *Journal of Digital Imaging*. 22(6):605–619, 2009
6. Maani R, Camorlinga S, Eskicioglu R: A remote real-time PACS-based platform for medical imaging telemedicine, *Medical Imaging 2009*. In: Siddiqui, KM, Liu, BJ Eds. *Advanced PACS-based Imaging Informatics and Therapeutic Applications*. Proceedings of the SPIE. SPIE Press, Orlando, USA, 2009, vol 7264, pp 72640Q-72640Q-12
7. Ramakrishnan B, Siraam N: Internet transmission of DICOM images with effective low bandwidth utilization. *Digital Signal Processing*. 16(6):825–831, 2006
8. Clunie DA: Lossless compression of grayscale medical images: Effectiveness of traditional and state-of-the-art approaches. In: Blaine GJ, Siegel EL, Eds. *Medical Imaging 2000: PACS Design and Evaluation: Engineering and Clinical Issues*. Proceedings of SPIE. SPIE Press, Bellingham, WA, 2000, pp 74–84.
9. Foos DH, Muka E, Slone RM, Erickson BJ, Flynn MJ, Clunie DA, Hildebrand L, Kohm KS, Young SS: *JPEG 2000*

- compression of medical imagery, Medical Imaging 2000: PACS Design and Evaluation: Engineering and Clinical Issues. 2000, pp. 85–96
10. Kalyanpur A, Neklesa VP, Taylor CR, Daftary AR, Brink JA: Evaluation of JPEG and wavelet compression of body CT images for direct digital teleradiologic transmission. *Radiology*. 217(3):772–779, 2000
11. Sung MM, Kim HJ, Yoo SK, Choi BW, Nam JE, Kim HS, Lee JH, Yoo HS: Clinical evaluation of compression ratios using JPEG2000 on computed radiography chest images. *Journal of Digital Imaging: The Official Journal of the Society for Computer Applications in Radiology*. 15(2):78–83, 2002
12. Erickson BJ: Irreversible compression of medical images. *Journal of Digital Imaging: The Official Journal of the Society for Computer Applications in Radiology*. 15(1):5–14, 2002
13. Amor NB, Essoukri N and Amara B: DICOM image compression by wavelet transform. In: *Systems, Man and Cybernetics*, 2002 IEEE International Conference on, 2002, vol. 2, pp 553–557
14. Seeram E: Irreversible compression in digital radiology. A literature review. *Radiography*. 12(1):45–59, 2006
15. Sung M-M, Kim H-J, Kim E-K, Kwak J-Y, Yoo J-K, Yoo H-S: Clinical evaluation of JPEG2000 compression for digital mammography. *IEEE Transactions on Nuclear Science*. 49(3):827–832, 2002
16. Logeswaran R: *Compression of medical images for teleradiology, Teleradiology*. Springer, Berlin Heidelberg, 2008, pp 21–31
17. Pianykh OS: *Digital imaging and communications in medicine: A practical introduction and survival guide*. Springer Publishing Company, Incorporated, Leipzig, Germany, 2008
18. Maani R, Camorlinga S, Arnason N, Eskicioglu R: *A practical fast method for medical imaging transmission based on the DICOM protocol*, Medical Imaging 2010. Advanced PACS-based Imaging Informatics and Therapeutic Applications, San Diego, USA, 2010, p 76280M
19. National Electrical Manufacturers Association, DICOM Official Website. <http://medical.nema.org/>. Accessed in December 2009.
20. Vossberg M, Tolxdorff T, Krefting D: DICOM image communication in globus-based medical grids. *Information Technology in Biomedicine, IEEE Transactions on* 12(2):145–153, 2008
21. DCM4CHEE, Open Source Clinical Data Manager system. <http://www.dcm4che.org/confluence/display/ee2/Home>. Accessed in December 2009.
22. Zimmermann H: OSI reference model-the ISO model of architecture for open systems interconnection. *IEEE Transactions on Communications* 28(4):425–432, 1980
23. Forouzan BA, Fegan SC: *Data communications and networking*. McGraw-Hill Professional, New York, NY, 2003
24. OsiriX Imaging Software, DICOM sample image sets. <http://pubimage.hcuge.ch:8080/>. Accessed in January 2010.
25. National Electrical Manufacturers Association (NEMA), NEMA DICOM CD 1997. <ftp://dicom.offis.de/pub/dicom/images/>. Accessed in January 2010.
26. eFile Community, DICOM Sample Images. <http://www.merge.com/efilmcommunity/sample.htm>. Accessed in January 2010.