ISSN 1870-4069

Collaborative Multi-Source Scheme for Multimedia Content Distribution

Javier Mendoza Almanza, Francisco de Asís López-Fuentes

Universidad Autónoma Metropolitana-Cuajimalpa,
Departament of Information Technology, Mexico City,
Mexico

flopez@correo.cua.uam.mx

Abstract. Demand for multimedia contents has increased in recent years, and several distribution services have emerged. Many of these multimedia distribution services are based on central servers, which introduce several limitations related with costs, dependence, performance or scalability. This paper presents a collaborative scheme for multimedia content distribution. Collaborative infrastructures for multimedia services are critical because multimedia contents have an import consume of resources in the communication networks. P2P networks have emerged as promising solution to implement collaborative infrastructures. Multi-source schemes are a practical solution when different parts of multimedia content is generated or stored in two or more sites. We have used a P2P network to implement a practical distribution prototype of our collaborative multi-source scheme. Our evaluation shows as peers share storage capacity, contents and bandwidth capacity, while server is released from this workload.

Keywords: P2P networks, content distribution, distributed systems.

1 Introduction

During recent years, several content distribution infrastructures have emerged in response to high demand for multimedia contents. Most of these infrastructures have based on central servers, which present several limitations and present a reduced collaboration between requesting nodes. Because the multimedia content consumes a large amount of resources in a communication system, collaboration between requesting nodes play an important role. In this context, peer-to-peer (P2P) networks have emerged as practical solution for constructing collaborative infrastructures. P2P systems have generated great interest in the research community who find in these systems a fast and efficient way to deliver movies, music or software files [1, 14, 15]. In a P2P system, the users interact directly as a way to exchange their resources and services through the Internet. Multimedia content requires large storage spaces, and large multimedia contents (e.g. movies) often exceed the storage capacity of a personal device. Multi-source schemes are used in order to solve these requirements. Multi-source schemes are also required when content is generated from multiples sites. For

example, in soccer match. This paper reviews different content distribution schemes, and introduces a collaborative distribution scheme based on P2P networks. In a P2P network, a node can take the role of both a server and of a client at the same time. Thus, when a new peer arrives to the system, the demand in the system is increased, but the system's overall capacity is increased too. A P2P system distributes its load and duties between all participating peers, which is not possible in a system based on central servers.

P2P networks are becoming more and more popular today (they already generate most of the traffic in the Internet). For instance, P2P systems are very used for file sharing and distribution; some examples are Bittorrent [2], Tribbler [3], eMule [4], GridCast [11], etc. Main technical problem is that peers connect and disconnect with high frequencies, in an autonomous and completely asynchronous way. This means that the resources of the network as a whole are also highly variable, and thus, that the network must be robust face to these fluctuations. In order to face the high dynamicity of such a system, we explore a multi-path approach where (i) the stream is decomposed in some way into several flows; (ii) each client receives those flows following different paths and sent from different other clients.

On the other hand, in a multi-source scheme, each source can distribute different video sequences to all requesting peers. How much the source can redistribute depends on the available upload capacity. At the same time, each requesting peer forwards the video directly received from a source to the rest of the peers. Again, the amount of redistributed content depends on the peers upload capacity. The upload capacities of the sources are divided equally among the different video streams. In this paper, we present a collaborative multisource scheme based on P2P networks. In this paper is proposed a multi-source scheme to disseminate multimedia content from multiple source to multiple requesting peers. Initially, sources distribute the original content to each requesting peer. After the requesting peers receive the content, they can redistribute this content to other peers in a collaborative way.

The rest of this paper has the following organization. In Section 2, we present information about different multi-source schemes based on P2P networks. Section 3 presents our proposed model. A practical implementation of our multi-source scheme is described in Section 4. Our paper concludes in Section 5.

2 Related Work

Dissemination information to a large group of nodes from many sources is fundamental in many systems and applications. Multi-source P2P multicast applications recently have been used for collaborative environments such as conferencing or multi-player games. A P2P network is an overlay network formed by a group of nodes. P2P systems maintain their independence of the underlying physical network by using this overlay topology. In a P2P network, a company can disseminate information. Thus, content can be distributed to an audience without the need for any special support from the network (Jannotti et al 2000), and where the upload capacity of the participating peers is only considered to forward the content. We can create a collective organizational knowledge within the organization, or share data and application files between computers without a dedicated server. However, P2P overlays known as unstructured

and structured overlay show limitations for multi-source multicast such as scalability [6], large overhead [7] or complex protocols [8].

Several approaches for content distribution from multiple sources to a single receiver can be found in the literature. Authors in [9] exploit the similar source concept to significantly improve the download time of a file from multiple sources to one receiver. Push is proposed in [10] as an efficient generic push-pull dissemination protocol. Pulp exploits the efficiency of push and pull approaches, such that it presents achieve reasonable latency and presents a low overhead by limiting redundant messages. In a multi-source environment, sources can provide conflicting values (false or true information). To deal with this problem, authors in [12] propose Datafusion, a novel solution to find true values from conflicting information when in the system there are a large number of information sources. In [13] is proposed a framework for video delivery from multiple sources to multiple receivers using P2P networks. In this work, authors consider that sources are requesting peers too. This work introduces a concept called helper peer, which is not interested in receiving the content and just contribute their resources during distribution.

3 Proposed Model

Our proposed model with multiple sources is shown in Figure 1. Each source distributes its initial content to different requesting peers. Our solution assumes that sources are independent of each other. Different type of files are distributed from each source. Source S1 distributes video files, while S2 distributes music files, S3 distributes photos files and S4 distributes PDF files. S5 is used to distribute other type of files.

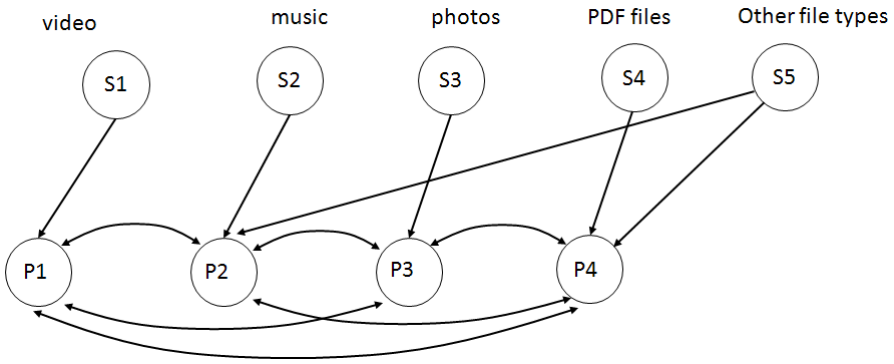

Fig. 1. Proposed model.

Initially, each server sends its content to requesting peer. After a peer receives a type of file, it can be distributed to rest of requesting peers. Thus, each source distributes only its files in the first stage. In the second stage, files are redistributed among all requesting peers. Each server has two functions. First function is to distribute the original content, and second function is to maintain information about peers with distributed files. This information is stored in a database in each server, which is periodically updated. In this way, a server reduces its workload. When new peers arrive

to system, information about IP address of each peer and its shared content can be obtained from any server.

In a multi-thread application, different processes can be executed at the same time concurrently. The number of processes is variable and depends on the amount of connections that are established. In this work, we have used multi-thread to improve the performance of our system. Figure 2 shows threads active in each peer. Coordinator peer is the main thread in each peer, and it is responsible for synchronization and control of the others threads. Client_thread_1 is responsible for connecting with the main servers. Thus, this thread requests a content and receive it from the source. Client_thread_1 also receives from the source the table with information of IP address and name (ID) of all requesting peer in the system. In this way, if a requested content is available in a requesting peer, then the content is downloaded from this peer. Client_thread_2 is responsible for receiving content from one or more peers in the system. To request contents from other peers, each peer uses the database with IP address and name of peers. A peer can receive multimedia contents from the main sources and from other requesting peers. These contents are stored in a database. Using this information each peer can work as server and redistribute information to other peers. Server_thread is responsible of this task in each peer.

4 Implementation

We put in practice our scheme by using different entities and servers, these entities are peers and servers. The servers store information about the connected peers and the file shared in each peer. There are different dedicated servers and each of them gives out and stores information about different types of files, which means there is a server for video another one for images another for music, another for pdf, and another one for any other kind of file so that we can do it multi-source. In this work, two main

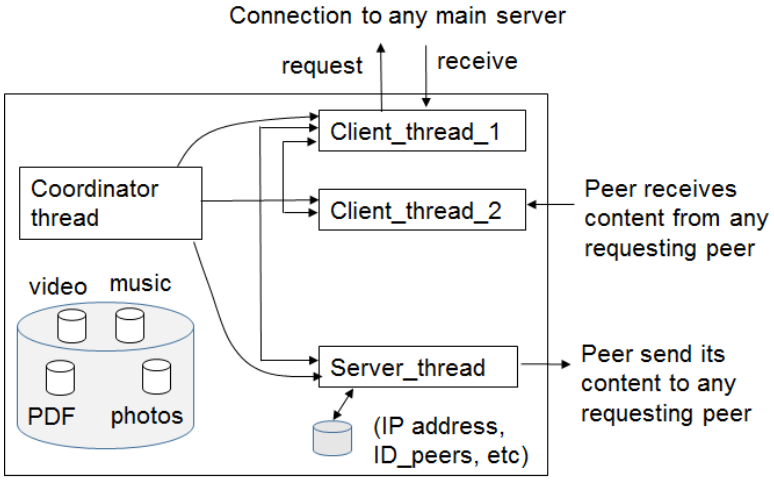

Fig. 2. Peer model.

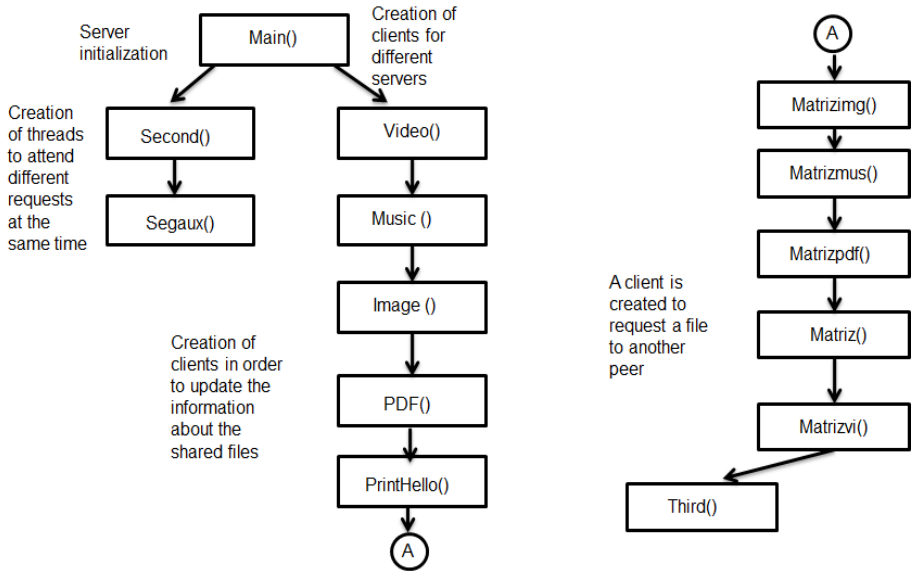

Fig. 3. Flow diagram for peer application.

applications called peer and server has been implemented. Each node in the P2P network runs a peer application, such that each node must receive and send files at the same time. To reach time goal, peer application performs both tasks simultaneously. Peer application is formed by two parts: a server and a client. Server part always is listening in order to attend to other peers, while client part makes different functions such as uploading files, display files and exit. Peer application is placed in each node of the P2P network (in each individual computer). A peer is an entity that sends and receives files at the same time. Peer application creates different threads when is running. First thread is to manipulate the server, and then other threads are created to connect them to the different servers. Each peer inserts its IP address, the amount of shared files with their names. Each server sends to all peers the information about the different connected peers. Each server sends updated information about the connected peer to all peers in the system. Figure 3 shows the flow diagram for peer application. Here, we can see the different steps developed by this application.

Our second application is the server. This application is responsible to give information about the peers in the system and store all files to be shared. A server is receiving requesting from peers while is sending files to them. While a server application is running, a socket is listening and waiting for new requests. When a new request from a peer is received in the server, a new thread is created to attend this request. Each server has a global matrix where IP address and the files names that peers want to share are registered. For each file to be shared by a server is created a thread toward that peer in order to store that file. Peer receives the matrix with information of connected peers and the shared files by these peers. An experimental prototype is implemented using five different servers. Each server offers a dedicated service, such that each server manages a specific type of file: video, music, images, PDF and one for other kind of files. Main steps for server application is shown in Figure 4. A matrix is

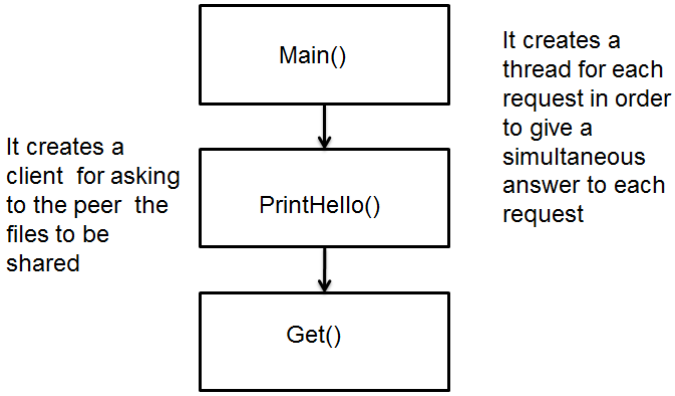

Fig. 4. Flow diagram for server application.

used to store the name of the shared files and the IP address of the peers in order to request files to different peers.

We have evaluated the operation of our prototype in a local red of our campus. Both servers and peers work correctly. Files are sent from each dedicated server and each requesting peer correctly, and after this, each peer can forward the received files to rest of peers. To compare performance of our collaborative P2P scheme against a distribution scheme based on client-server, we have measured the distribution time to requesting peers. Our work is in progress, and preliminary tests have been done. First, we distribute four files of 28MB to two clients from a server at same time. Client 1 receives the four files from the source in 7.56 minutes, while client 2 receives the four files from the source in 8.08 minutes. On the other hand, using P2P architecture, peer 1 receives the four requested files in 2.25 minutes, while peer 2 receives the four files in 2.16 minutes. Preliminary results show that P2P architecture presents a best performance than architecture based on client-server because. This is because the server is congested to send all files, while in the P2P scheme all nodes collaborate to distribute content faster. However, obtained measurements may change depending on the variation of the network. We can continue testing our collaborative transmission scheme in order to make more efficient our proposal.

5 Conclusions

There is currently a high demand for multimedia content, and collaboration among requesting nodes play an important role. In this paper has been proposed a collaborative multi-source scheme to distribute multimedia content to many requesting peers. Collaboration between peers is implemented by using a peer-to-peer network. Our framework is suited for collaborative environments, where the system inherently has multiple senders. Using this proposed approach, the sources can distribute their workload between all requesting peers, and the system can improve its performance. Our collaborative scheme uses threads to establish collaboration connections with other peers. The number of threads is variables and depends on the amount of established

connections. In each peer, a coordinator thread deals with the incoming requests and creates the rest of threads that handle the requests. Our current effort is focused to complete the implementation of our proposed framework for video streaming sessions.

References

1. Milojicic, D., Halogeraki, V., Lukose, R., Nagaraja, K., Pruyne, J., Richard, B., Rolling, S., Xu, Z.: Peer-to-Peer Computing. HP Labs Technical Report HPL, 57 (2002)
2. Cohen, B.: Incentives to build robustness in BitTorrent. Technical report, http://www.bittorrent. com/bittorrentecon.pdf (2003)
3. Pouwelse, A., Garbacki, P., Wang, J., Bakker, A., Yang, J., Iosup, A., Epema, D. H., Reinders, M., Van Steen, M. R., Sips, H.: TRIBLER: a social-based peer-to-peer system. Journal Concurrency and Computation: Practice & Experience, Vol. 20, No. 2, pp. 127–138 (2008)
4. The eMule Project: https://www.emule-project.net (2016)
5. Jannotti, J., Gifford, D. K., Johnson, K. L., Kaashoek, M. F., O'Toole Jr., J. W.: Overcast: Reliable Multicasting with an Overlay Network. In: Proc. of the 4th Symposium on Operating System Design and Implementation (OSDI'00), San Diego, CA, USA, pp. 197–212 (2000)
6. Chawathe, Y.: Scattercast: An Adaptable Broadcast Distribution Framework. In: ACM Multimedia Systems Journal Special Issue on Multimedia Distribution (2002)
7. Ripeanu, M., Foster, I., Iamnitchi, A.: Mapping the Gnutella Network: Properties of Large-Scale Peer-to-Peer Systems and Implications for System Design. Internet Computing Journal, Vol. 6 (2002)
8. Bharambe, A. R., Rao, S. G., Padmanabhan, V. N., Seshan, S., Zhang, H.: The Impact of Heterogeneous Bandwidth Constraints on DHT Based Multicast Protocols. In: Proc. of the 4th International Workshop IPTPS (2005)
9. Pucha, H., Andersen, D. G., Kaminsky, M.: Exploiting Similarity for Multi-Source Downloads Using File Handprints. In: Proc. of the 4th USENIX NSDI 07, Cambridge, MA, USA (2007)
10. Felber, P., Kermarrec, A. M., Leonini, L., Riviere, E., Voulgaris, S.: Pulp: An adaptive gossip-based dissemination protocol for multisource message streams. Peer-to-Peer Networking and Applications, Springer US (2012)
11. Cheng, B., Stein, L., Jin, H., Liao, X., Zhang, Z.: Gridcast: improving peer sharing for P2P VoD. In: ACM TOMCCAP (2008)
12. Dong, X. L., Berti-Equille, L., Srivastava, D.: Data fusion: resolving conflicts from multiple sources. Handbook of Data Quality, pp. 293–318 (2013)
13. López, F. A., Steinbach, E.: Multi-source video multicast in peer-to-peer networks. In: Proc. of the IEEE International Symposium on Parallel and Distributed Processing, Miami, FL, USA, pp. 1–8 (2008)
14. Sayit, M., Demirci, S., Kaymak, Y., Tunali, E.: Adaptive, incentive and scalable dynamic tree overlay for p2p live video streaming. In: Peer-to-Peer Networking and Applications, pp. 1–15 (2015)
15. Kuo, J. L., Shih, C. H., Ho, C. Y., Chen, Y. C.: Advanced bootstrap and adjusted bandwidth for content distribution in peer-to-peer live streaming. In: Peer-to-Peer Networking and Applications, pp 1–18 (2014)