Peer-to-Peer Simulation for Improving the System of Serving Multimedia on the Web

Maryam Bashardoust
Electrical and Computer Eng Dept.
Ryerson University, CA
mbashard@ryerson.ca

Martin Gerdzhev
Electrical and Computer Eng Dept.
Ryerson University, CA
mgerdzhe@ryerson.ca

Abdolreza Abhari
Computer Science Dept.
Ryerson University, CA
aabhari@scs.ryerson.ca

Keywords
Peer-to-Peer, Client-Server, multimedia, BitTorrent, latency, YouTube, simulator

Abstract

P2P networks are scalable in serving multimedia files over Internet which is in contrast to the centralized client-server system in which the server provides all the resources to the clients. In this research, we have designed and implemented a Peer-to-Peer network simulator to efficiently serve multimedia on the Internet considering that P2P streaming applications scale very superior even in large crowd scenarios. Simulated data was used to conduct experiment from current popular multimedia serving websites such as YouTube. We have then simulated user traffic by using the P2P system to compare the performance result with the traditional Client-Server system.

1. Introduction

Peer-to-peer (P2P) topology utilizes various connections among the computer nodes in a network and enhances the efficiency usage of the processing power and disk space. In P2P networks, adding more peers to the system increases the total available resources and the capacity of the system, since every node provides bandwidth, computing power and storage space.

P2P technology offers wide range of facility for users including multimedia streaming, information retrieval, online gaming, and conferencing. However P2P is highly demanded for its file sharing applications over Internet such as BitTorrent. In file sharing applications based on P2P topology, files are stored and served over the network among users. This file sharing system requires a balance between uploading and downloading files as opposed to file trading which operates around downloading files from a P2P network without the necessity of uploading.

Recent researches and studies show the significant improvement in the performance of multimedia delivery by adopting P2P topology rather than the traditional client-server architecture. In view of the fact that in client-server architecture, the limited bandwidth and disk space related to the server is problematic. The increase in the number of simultaneous client requests to handle by the server leads to diminishing the number of available resources on the server side. As a result, server becomes strictly overloaded and traffic congestion on the network may occur. P2P has the potential of a reliable architecture to prevent a similar problem, since the available resources grow as the clients' requests increase.

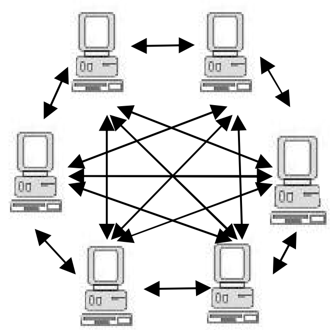

Figure 2. Peer-to-Peer

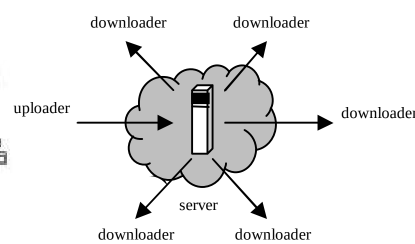

Figure 1. Client-Server Architecture

2. Related Work

The related works and studies in this area demonstrate the importance of adopting P2P topology to serve today's Internet users. Li et al.'s research [1] presents that P2P networks usage is on the rise due to the reliable characteristics of P2P system in contrast to the centralized client-server. P2P networks are utilized for multimedia streaming, file backup, and etc; however the major role of P2P is providing file sharing applications for users such as BitTorrent, Skype, and Napster. Later the writers continue exploring characteristics of BitTorrent which is an opensource P2P file sharing protocol and Skype which is an internet telephony network (VoIP).

Wu et al. [2] reveal that real-world live P2P streaming applications are recently emerged successes that are able to serve users adequately by having simplicity in design regarding to topology construction protocols selection and with less sophisticated algorithms. The system design includes a live media stream which is divided into blocks and is delivered over a mesh overlay topology in each session and periodically exchange useful blocks of data among peers in the network.

Kalogeraki et al. [3] define the P2P networks problem as the centralized indexing methods used in early P2P systems being very restrictive seeing as when groups of sites tend to form sizable node clusters, the large number of dispatched messages causes inefficient use of computing resources. The proposed architecture to resolve this problem is an approach to ad-hoc P2P networks of resources to support storage and retrieval of multimedia objects by using partial and exclusive indexing in peers. This proposed architecture guarantees continuous functioning in case of server light failures.

Lan et al. [4] propose a manageable overlay P2P network for efficient scalable live media streaming (coded media by scalable coding scheme), in which every peer periodically exchanges its data availability information with one of the distributed central servers. In addition, an efficient scheduling algorithm is used to achieve continuous and real-time transmission of the scalable streaming.

The last related work that we are going to examine is a research done by Microsoft. Padmanabhan et al. [5] have addressed the problem of live streaming media content (simultaneous and real-time distribution of the similar content) to a large and dynamic number of clients by using P2P network, however the challenging point is making content distribution robust to peer transience. The writers proposed solution is using redundancy in data by using multiple description coding (MDC) and redundancy in network paths by using multiple diverse distribution trees.

3. Simulation

This section contains all the detailed and comprehensive information about the Peer-to-Peer simulation for proficient multimedia delivery system on the Internet. For the implementation of the P2P network simulator the language used Java 6.0 due to its portability, ease of use and extensibility. The development was done using the Eclipse IDE for its ease of use.

Our designed P2P simulator takes a file with client requests for specific videos at specific times as an input. The timing information is generated using a Poisson distribution random number generator in which differences of the values are taken and put as the next access time after the current access. The next access time will be calculated after finishing the video and each subsequent access continues from that point. This ensures that there are no incomplete video downloads in the simulation. The output file of this P2P simulator contains information such as the number of clients connected to the server, the server's total and per client upload speeds, It also contains the total number of clients that have requested a video up to that point, clients' total download speed and average per client download speed. Other information is also provided, such as the average number of clients per video and the number of clients that are downloading a video that other peers have already downloaded or are downloading at the moment. These statistics are output every second.

In the proposed P2P structure, the server contains all the information about the videos and the peers and also acts as a seeder for all the videos. Also it should be noted that the designed simulator doesn't take into account any network transmission latencies or time required to establish connections to other machines as they would only have a minimal impact.

With respect to the server and the client diverse capabilities, the maximum upload speed for the server and all the clients is adjustable. The P2P simulator consists of two tabs, the server tab and the client tab, which show real time information about the server and all the clients. The server tab contains statistics for each video that the server has. These statistics display video id, file size, file duration, and upload speed of the video. Moreover, they show the total number of peers connected to the server, the average upload speed to each peer and the total upload speed of the server. The client tab contains statistics for each peer. These statistics include client id, video id of playing video, size of video, percentage of video downloaded, current time in video, duration of video, and download speed and upload speed of the client. The statistics also show the total number of peers connected to the present client peer, the average and the total download speed, the average and the total upload speed for all client peers.

The proposed Peer-to-Peer simulator is based on the BitTorrent protocol where peers upload at the same time as they download. In the simulated P2P network, multiple peers are connected to one peer, its upload bandwidth gets split equally among all the connected peers that need fragments from that peer. In general, peers download from other peers in the network that have the required fragments they need. These fragments are downloaded randomly within a specified buffer of missing fragments to ensure that most of the newly downloaded fragments are close to the beginning of the file. This process is to avoid the complete incremental download in which case when all the peers have the same data, as a result they only can download from the server, and to ensure a smooth playback without interruptions.

After a file is downloaded, it remains in the peer's list as a file being seeded. In this way, when a client peer finishes one video and starts watching a different video, the first finished video will stay in the peer's list. Hence, as another peer requests the same buffered video, it is able to download it from the first peer along with other requesting peer nodes.

4. Results

For the simulation we used 200 Clients, each having 6 requests, totaling 1200 requests for videos. There were 683 unique videos with an average file size of 3.8 MB. A distribution simulator software[6] was used to generate files with client requests for popular videos as they are downloaded more frequently and the biggest benefit would be seen for them. These files were then merged into 1 file of client requests and timing information was added to it as well. Then this final file was used as an input file for our Peer-to-Peer simulator. The simulation completed in 6 hours. It should be considered that some of the videos were requested multiple times, and others just once. This simulation had the settings for maximum server upload speed of 2 MB/s from which each client would receive maximum 200 KB/s and maximum client upload speed of 40 KB. A limit to the number of connections of a client was set to 50 and a maximum download speed for each client was set to 520 KB/S to account for an average user's bandwidth limitations.

The performance metrics used are the server's total upload speed, server's upload speed per client which is total upload speed divided by the number of clients connected to the server. Other performance metrics used are the clients total download speed which is the sum of download speeds of all the clients, and the average download speed per client which is the total download speed divided by the number of clients that are currently downloading. The difference between what the server provides and the client actually gets is the bandwidth that is provided by the other clients. This is important because we can determine by how much the load on the server can be reduced with this architecture.

During the process of the simulation we observed spikes in the total download speed of the peers. These spikes occurred when one or more peers requested a video that was either currently being downloaded by other peers or had been previously downloaded by other peers.

From our simulation it can be observed that on average, the total download speed of all the clients was about 9% higher than the upload speed of the server. The same can be seen for the average download speed per client which was 9% faster than the upload speed per client of the server (See Figure 3). Other average statistics are shown in Table 1.

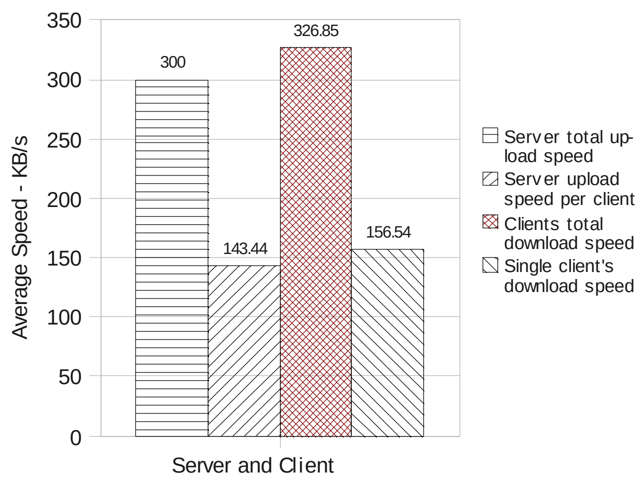

Figure 3. Average Server Upload and Average Client Download

Table 1. Average statistics for simulation

Number of peers connected to server	1.05
Server total upload speed	300 KB/s
Server upload speed per client	143.44 KB/s
Total number of peers	168.89
Clients total download speed	326.85 KB/s
Average download speed per client	156.54 KB/s
Number of clients downloading previous video	0.27

During the peak where the Single client's download speed is maximum, it can be observed that 1 peer is currently downloading the video from at several more peers other than the server. For that reason, the client's download speed in this moment is 2.6 times greater than the upload speed of the server. (See Figure 4 and Table 2)

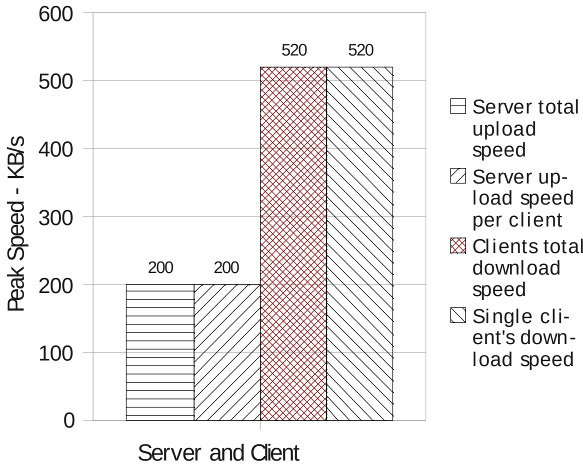

Figure 4. Peak Server Upload and Peak Client Download

Table 2. Statistics at peak client download speed

Number of peers connected to server	1
Server total upload speed	200 KB/s
Server upload speed per client	200 KB/s
Total number of peers	21
Clients total download speed	520 KB/s
Average download speed per client	520 KB/s
Number of clients downloading previous video	1

What can be observed from this is as more clients download videos, there is more chance for other clients to download the same video and thus reduce the load on the server substantially.

When all the clients were downloading videos that had not been downloaded before, the upload speed of the server matched the download speed of the clients as the clients were only downloading from the server.

5. Limitations and Future Work

At its current state, the simulator doesn't take into account any network transmission latencies or time required to establish connections to other machines. It needs to be mentioned that improving these areas would give more accurate results.

Our test scenario included a very small set of data. A future series of tests should be carried out which should include more clients and more videos. Various maximum server upload and client upload speeds should also be tested.

6. Conclusion

Based on the experiment and the results gained from the simulated traffic using P2P system, we can conclude that for our setup the Peer-to-Peer model on average 9% of the bandwidth was shifted to the peers as opposed to the traditional client-server model where the server takes the full load. It is observable that when the number of clients requested a video which was either currently being downloaded by other peers or had been previously downloaded by other peers increases, the average download speed per client increases as well.

Peer-to-Peer model is beneficial for networks with servers that handle a large volume of multimedia files and have high number of users such as YouTube. The results demonstrate that when the number of peers that download the previously downloaded videos increases, so does the bandwidth at no extra cost for the server. This could potentially solve any scalability problems that multimedia serving companies might face due to an increase in the number of users of the servers.

References

[1] Li, J. *Peer-to-Peer multimedia Applications.* International Multimedia Conference, Proceedings of the 14th annual ACM international conference on Multimedia. ACM, New York, USA. 2006.

[2] Wu, C., Li, B., AND Zhao, S. *Exploring Large-Scale Peer-to-Peer Live Streaming Topologies.* ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP). ACM, New York, USA. 2008.

[3] Kalogeraki, V., Delis, A., Gunopulos, D. *Handling Multimedia Objects in Peer-to-Peer Networks.* Proceedings of the 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid. IEEE Computer Society, Washington, DC, USA. 2002.

[4] Lan, X., Zheng, N., Xue, J., Wu, X., Gao, B. *A Peer-to-Peer Architecture for Efficient Live Scalable Media Streaming on Internet.* International Multimedia Conference, Proceedings of the 15th international conference on Multimedia. ACM, New York, NY, USA. 2007.

[5] Padmanabhan, V.N., Wang, H.J., Chou, P.A. *Resilient Peer-to-Peer Streaming.* Proceedings of the 11th IEEE International Conference on Network Protocols. IEEE Computer Society, Atlanta, GA, USA. 2003.

[6] M. Soraya, M Zamani,A. Abhari, “Modeling of Multimedia Files on the Web 2.0'”, 21th annual IEEE Canadian Conference on Electrical and Computer Engineering 2008(CCECE'08): Symposium on Computer Systems and Applications, Niagara Falls, Canada, May 2008