Peer to Peer Video on Demand Architecture using V-Chaining

Hareesh.K
Research scholar
Jawaharlal Nehru Technological University,
Anantapur, A.P India

Manjaiah D.H
Professor & Chairman
Department of Computer Sciences,
Mangalore University, Mangalore

ABSTRACT
In the Internet, video streaming requires greater amount of network bandwidth and other resources as the number of user requests increases. In case of traditional centralized directory server approach all the users requests are directly handled by the centralized server and each user request will send dedicated stream by the server, which requires higher end server, server cost will become more and greater amount of network bandwidth utilized by this server. To solve these problems peer to peer technology as emerged for the distribution of video streams to the larger requests over the network. In P2P VoD architecture adopted both the peer to peer and proxy based architectural design of a VOD system for larger community of users over the network. Hence our proposed Peer to Peer Video on Demand Architecture using V-Chaining improves the overall performance of the system by efficient utilization of uplink bandwidth and smaller amount of buffer space among the peers. In this paper we have introduce architecture for handle the large number of user requests over the communication network and ease of implementation.

Keywords
Video on demand (VoD), peer, bandwidth, buffer, video

1. INTRODUCTION
To meet the greater demands of growing multimedia applications, media streaming has been a research topic attract significant interests on the users over the past two decades. The goal of live video streaming is to satisfy the application requirements of as many users as much as possible, with limited server bandwidth and costs. The traditional client/server architecture requires the use of large higher end servers to maintain streaming to requested users at a large scale. The server's bandwidth costs increases rapidly as the number of user requests increases, and may not be manageable in large sectors with limited resources [1], IP multicast [2] and content delivery networks (CDNs)[3] attempted to endeavor the problem by consumes the resources in the edge or core routers, or by load balancing across a large number of edge servers. However, the problem of scalability to a large number of users in media streaming systems is only limited to a certain extent, not fully solved. Over the past few years, Peer-to-Peer (P2P) networks have emerged as a promising approach for distribution of multimedia data streams over a large scaled network which is included with VoD applications by utilizing the uplink bandwidth of peers [4-8]. P2P networks propose a different architectural design perspective; it guarantees that of less bandwidth utilization from dedicated streaming servers hosted by the service providers, while transmitting the multimedia data to the users when they serve data streams to each other. In most of the cases, the total capacity of bandwidth consumption is not reduced, even it is increased due to overhead of protocol induced messaging and redundancy.However the bandwidth consumption is distributed across the network among the peers, which can contribute, delay, and consume the data stream. Existing peer-to peer media streaming systems can be divided into two categories, live and on-demand media streaming, with the latter often referred to as video-on-demand (VoD) system. The peer to peer VoD system typically contains three components as shown in Figure.1. A single dedicated media server, which is of dedicated streaming servers that serve media content, and can be considered as a single dedicated media server. One or more a small number of index servers that keeps track of state information of the system, such as existing peers. It is often referred as a Proxy server. The Internet gateway will provides information about the media channels/Video data streaming. To further improve the server load of the VoD system, a client side caching scheme is proposed [9] called as earthworm. In this scheme the client not only playback the video but also forward the streams to another client with adequate buffer delay known as basic chaining. This scheme further extended as basic, standard, adaptive, optimal chaining which exploits client resources such as buffer and uplink bandwidth[10][11][12]. However demand for DVD quality videos and longer duration videos are expected in the near future [13]. For such applications the existing chaining schemes fall short in meeting scalability requirements like bandwidth and buffer. Due to the tremendous transfer of same data occurs in the VoD system which will degrade overall performance of the system. Hence our proposed Peer to Peer Video on Demand Architecture using V-Chaining improves the overall performance of the system and increase efficient bandwidth and buffer utilization.

In this paper, we have aim to design a novel P2P architecture both combined architecture of peer to peer and proxy based architectural design for the greater of community of users over the internet. We propose a robust and efficient P2P architecture using V-Chaining. It achieves the optimal network in a dynamic environment. The main contribution of this paper is we propose efficient P2P architecture using V-Chaining, which is an effective and robust.

2. CENTRALIZED ARCHITECTURE

Centralized directory server consists of Centralized media server in turn connected through fiber optic cables by the peers of the various clusters. Each cluster contains number of peers, which is directly request the video stream to the centralized media server. Each of video requested by the peers will directly handled by the server. The different parts of the peers are directly connected to the central part of the network through which access networks resources which may take multiple numbers of hops to reach the VOD server.

Figure 1 shows the centralized architecture with 4 clusters. Each Cluster consists of number of peers. The centralized directory server of the directory is maintained locating peers. The directory server maintains the required information about each peer such as address, available bandwidth, buffer, starting point of the location, IP address of the each peer is available in centralized directory server. The centralized directory server can also maintain the overlay structure among peers and above figure demonstrate the flow of information among peers.

Figure 2 shows the flow diagram for the centralized architecture. The Peers in turn are connected to the Internet Service Providers.

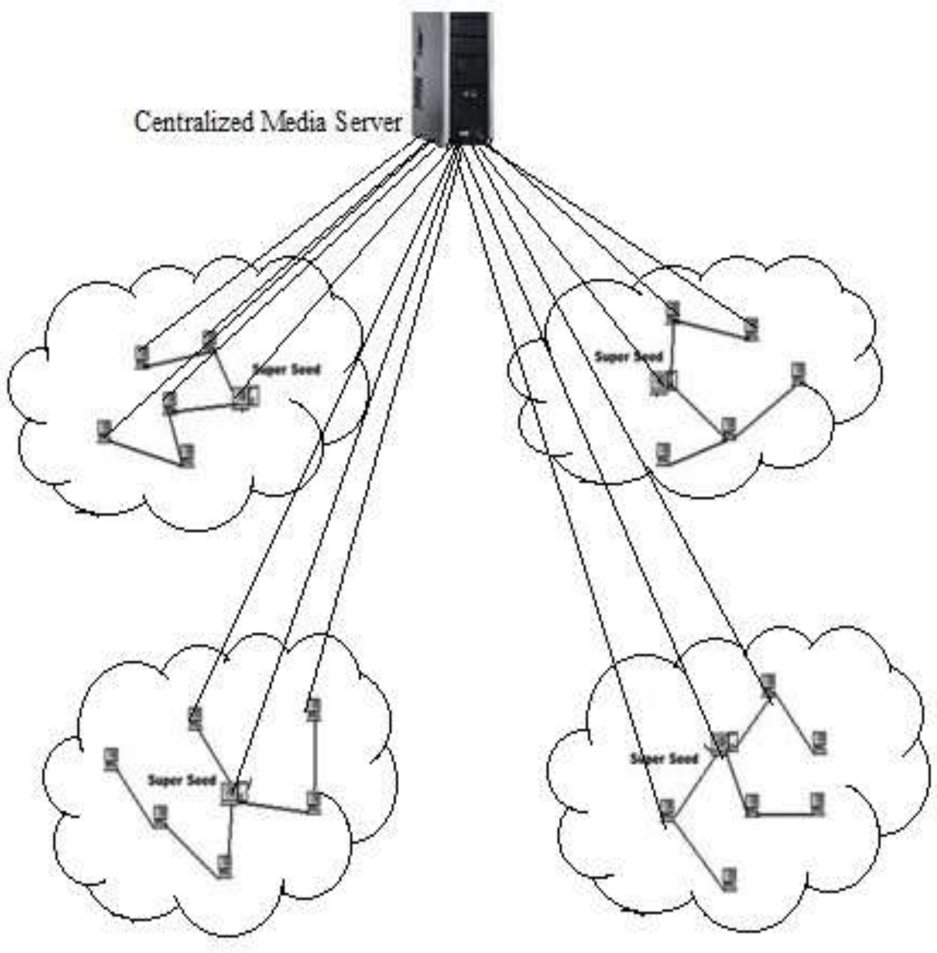

Figure 1.Centralized directory server architecture

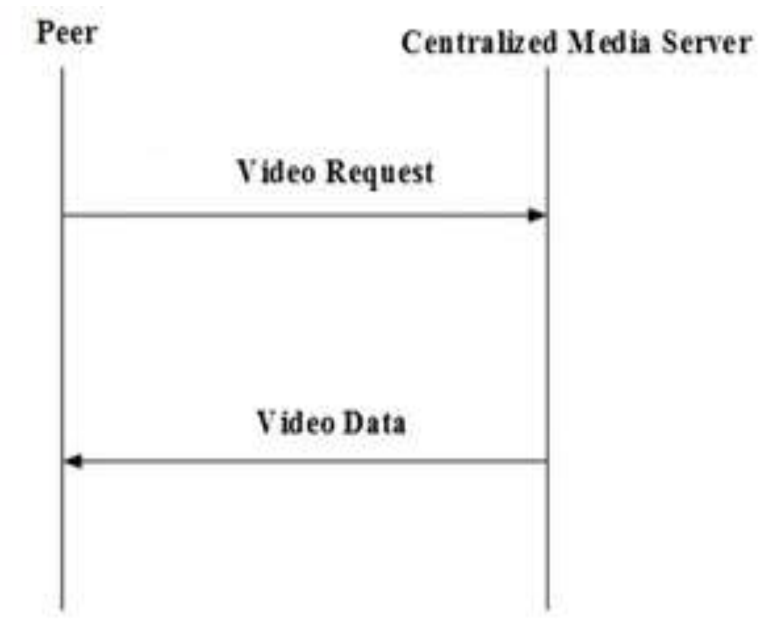

Figure2: Flow diagram for Centralized directory server

3. IMPLEMENTATION WORKING PROCESS

In this system, suppose if a new peer's request to the directory server then the requesting peer request is first directed to the directory server. Once receiving a the peer user request, the directory server will selects based on the network address and the requested media, the most suitable supplying peers from the stored peer list for the requested user. For example, to serve the requesting peer, a peer with large available bandwidth and a network access location which is very close to the requesting peer is chosen by the directory server. If a peer user wants to leave the system, LEAVE message information send to the directory server it indicates that peer may leave, this entry is also recorded in the directory. The advantages of this approach are simple deployment and ease of implementation. Centralization makes the join and leave procedures fast also greatly simplifies the join mechanism. However, given N peers in the system, the directory server must maintain $O(N)$ states, which will overload the server when the number N is large. Furthermore, if a peer is not able to send a LEAVE message to the directory server (that is during a node failure), its state remains in the directory.

To handle this problem, the peers must send periodically the *keep-alive* (or *heartbeat*) *messages* refresh their status at the directory server. The high-bandwidth consumption at the server continue transmitting these $O(N)$ keep-alive control messages by the peer. Another problem is that the directory server becomes a single point of failure. If the directory server is not working, users can no longer join the system. Hence, if the media server fails, it strength not matter whether the directory is itself not working.

4. PROPOSED VoD ARCHITECTURE

The peer to peer video on demand (P2P VoD) system is combined architecture of a media server, proxy servers and peers in a cluster. The media server contains a collection of video data files in its video directory. The video data file's information such as index, popularity, minimum bandwidth and minimum buffer, video length, timestamps information is also stored in the media server. In this architecture Proxy server is used to cache video data files for the nearer peers in a cluster. The purpose of proxy server is to reduce the load on the media server by caching the video data files. Each *Proxy server* has various modules, which operates at the time of streaming of videos are requested by the each of the peers in the cluster.

- *Communication Module of Proxy server:* communicate between one peer$_i$ with another peer$_j$.
- *Service Manager:* This module manages the currently streaming/streamed videos.
- *Request handler:* which handles the requests from the each of the peer which requested by the user to watch the movie.
- *Peer Manager :* monitor and updates the videos based on the popularity of videos at the *proxy server*
- *Chaining peers list:* which is the database contains the list of videos and the detail information of videos such as movie ID, index, popularity etc., and with entire information about list of videos being streaming/streamed from the proxy server and the consequent active chain of peers for that video.

- To each of these proxies a large number of peers are connected. Each proxy is called as a parent proxy to its peers. Each peer has various modules such as,
- *Buffer-manager:* buffers the video as and when the video segments received at the Buffer.
- *Media Player:* play back the buffered videos which are available in the buffer.
- *Peer Agent*: Performs the Communication with proxy server and other peers.
- *List of Peers:* which receive the peer information contains the video data from the proxy server.
- *Number of active Peers*: Number of the active chain of the video data, from requested peer can get the requested video stream.

The *proxy server* caches the prefix of the videos which are streamed from the media server, and then it streamed this cached portion of video to the requested peer through LAN, which is less expensive bandwidth. We have assumed that, proxies and their peers are closely located in the same region or cluster, which is quite low communication cost. The media server, in which all the videos completely stored is placed far away from proxy server, which involves high cost remote communication. The media server and the proxy server are assumed to be interconnected through high capacity fiber optic cables. A cluster is a logical connectivity of peers which is headed by a proxy server. A peer can be a seed peer or a non-seed peer. A seed peer is a client, which has sufficient bandwidth and buffer capacity as well as it can store and forward video data files to other peers. A non seed peer is a client, which has only required configuration and it can only playback the received video data file and cannot store or forward the video data files to other peers.

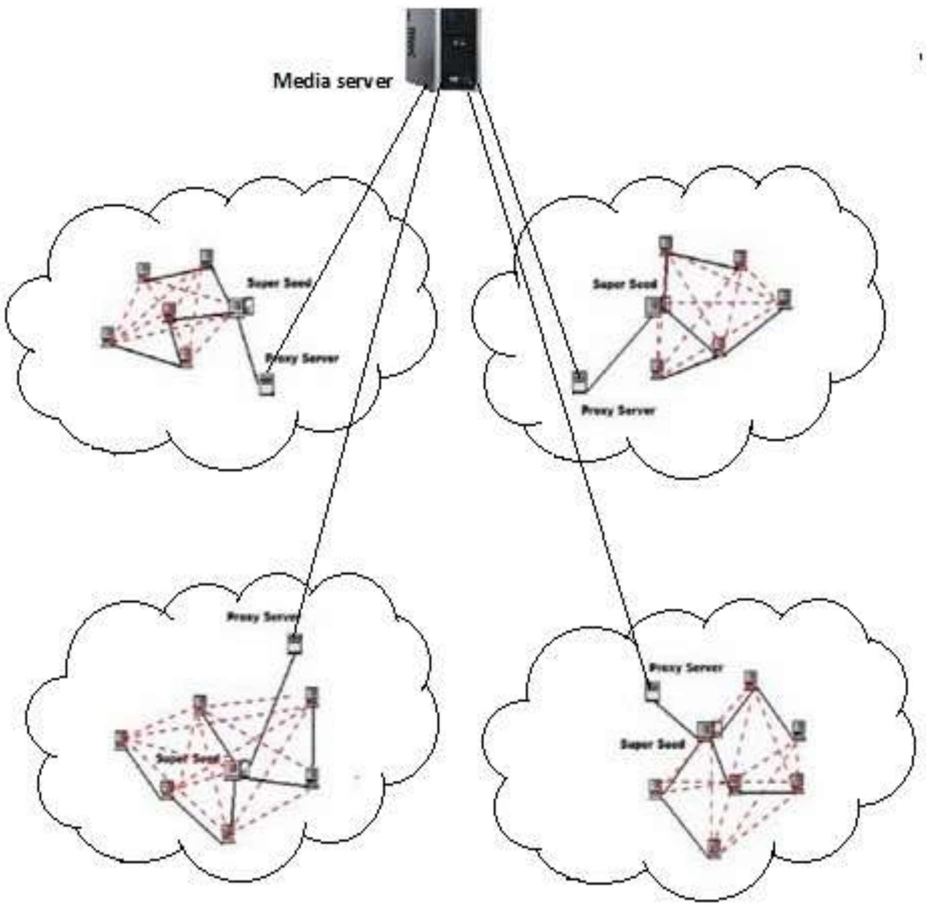

Figure 3: Proposed VoD architecture

As Figure shown in Figure 3, clusters are managed into multiple levels using distributed algorithms. Each cluster has a Super seed peer that is responsible for monitoring its cluster membership of the seed peer and is a member of a cluster in the upper level. Hence, some peers are Super seed peer in multiple levels. Cluster sizes can be between *k* and 4*k* (*k* is a system parameter) and are maintained using merge and split algorithms for bounding the out-degree of each peer. There's only one cluster in the topmost level where them source of the media server available. To join the system, a new peer first contacts the super seed of the topmost cluster in With that cluster's peer list attached in the Super seed peer will reply, the new joining peer which as new entry, which is measures the distances all the peers in the directory list and distance between neighboring peers. Then, it selects the closest seed peer, which is a super seed in the lower level. After that, the new peer sends another request to that super seed. Again, that closest super seed peer replies with its cluster member list. The inquiring process repeats until the new client finds its appropriate position in the architecture. By this successive inquiring, nearby seed peers are grouped together, making the data transmission based on that structure efficient. This approach by making use of the overlay structure shares the load among the various levels of the peers by the peer-management technique. For example: In case of centralized directory server of a directory maintains a $O(N)$ states, Each peer of Zigzag will also maintains $O(\log N)$ states. In case of Zigzag there is not problem of single point of failure because it maintains the overlay structure among the peers for this easy way of locating each of the peers. The control protocol of will send the heart beat messages to the server as and when the peer failure in the network and accordingly maintains the entire overlay structure. Because of frequent VCR-like operations, it is more challenging for VoD systems to maintain such a structural overlay for peer organization and data delivery.

Table1: Comparisons of architecture approach of Centralized directory and P2P VoD

Architecture Approach	Scalability and Dynamics	Single point of Failure	Server States	Peer States	Implementation
Centralized directory	Low	Yes	$O(N)$	$O(1)$	Simple
P2P VOD	High	No	$O(1)$	$O(\log N)$	difficult

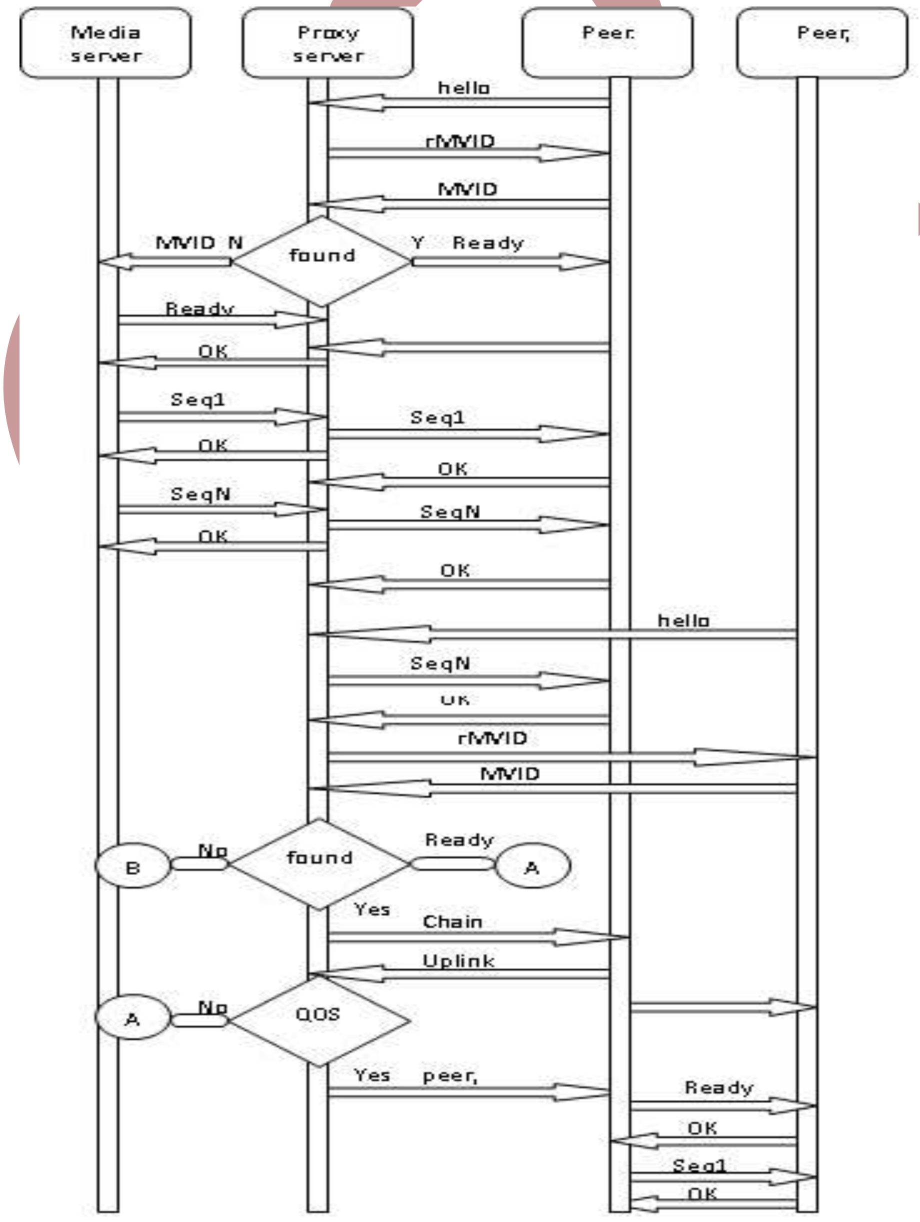

Figure 4 shows the flow diagram of VoD working process

5. IMPLEMENTATION WORKING PROCESS

Initially, suppose if a peer makes a request to the media server, and then the media server downloads the entire video data files to the nearest proxy server of the requesting peer. In the first case, it is assumed that none of the peers are requested for the same movie. Thereby, after downloading movie to the proxy server, the proxy server will transmit the movie to the requesting peer. Subsequently, if another peer from the same cluster makes a request for the same movie to the media server. Then the media server looks in to its current streaming movies database for the nearest proxy server and its availability of the movies. If such entry is found then media server redirects the requesting peer to the nearest proxy server. Again the proxy server applies the same procedure to find out that any of the peers in the cluster has the movie in its buffer. If such peer is found, then the requesting peer will be redirected to that peer which has the same movie. Then on the transmission occurs from that peer to the requesting peer. The transmission of video data file from that of peer to another peer is called *V-chaining*. However, if entry is not found in the proxy server, then the proxy server starts transmitting the video data files to the requesting peer. Elsewhere, if the entry is not found in the media server then the procedure is followed as if it is a first request from the cluster.

Suppose, if another request from different cluster occurs for the same movie to the media server. Then the media server redirects the nearest proxy server of the requesting peer to transmit the video data file to the proxy server of the requesting peer to transmit the video data file to the proxy server of the requesting peer. Now instead of downloading the video data file from media server to the nearest proxy server. The download happens from another proxy server which has the movie to the nearest proxy server of the requesting peer. Therefore, the same procedure is carried out for the transmission of movies among the clusters. If none of the proxy servers has the same movie then the media server downloads to the nearest proxy server and then the proxy server transmits the movie to the requesting peer.

6. CONCLUSION

In this paper, we have studied the comparisons of centralized directory server and peer to peer VoD architecture. In case of traditional centralized directory server approach all the users requests directly handled by the centralized server and each user request will send dedicated stream send by the server which requires higher end server, server cost will more and greater amount of network bandwidth utilized by this server. In P2P VoD architecture adopted both the peer to peer and proxy based architectural design of a VOD system for larger community of users over the network. The proposed architecture designed has the following benefits. The system architecture handles the larger number of user requests and efficient utilization of uplink bandwidth so that greater reduce of network bandwidth and also each video stream for temporary storage uses a smaller amount of buffer in each of the peer. We have used VChaining technique among the peers which will run the same stream over the longer duration which will benefit that same stream will not be used again and again. The access delay will greatly reduced using this architecture.

7. REFERENCES

[1] Y. Chu, S. Rao, and H. Zhang, “A Case for End System Multicast,” *Proc. ACM Sigmetrics*, ACM Press, 2000, pp. 1-12.

[2] K. Sripanidkulchai et al., “The Feasibility of Supporting Large-Scale Live Streaming Applications with Dynamic Application End-Points,” *Proc. ACM SIGCOMM*, ACM Press, 2004, pp. 107-120.

[3] Xuguang Lan, Nanning Zheng, Jianru Xue, Weike Chen, Bin Wang, and Wen Ma “Manageable Peer-to-Peer Architecture for Video-on-Demand,” *IEEE International Symposium on Parallel and Distributed Processing*, IPDPS 2008, April 2008.

[4] J. Li, “PeerStreaming: an on-demand peer-to-peer media streaming solution based on a receiver-driven streaming protocol,” in *Proceedings of the IEEE 7th Workshop on Multimedia Signal Processing (MMSP ’05)*, pp. 1–4, November 2005.

[5] H. Chi, Q. Zhang, J. Jia, and X. Shen, “Efficient search and scheduling in P2P-based media-on-demand streaming service,” *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 1, pp. 119–130, 2007.

[6] S. Annapureddy, C. Gkantsidis, and P. Rodriguez, “Providing video-on-demand using peer-to-peer networks,” in *Proceedings of the International Conference on World Wide Web*, 2006.

[7] C. Huang, J. Li, and K. W. Ross, “Peer-assisted VoD: making Internet video distribution cheap,” in *Proceedings of the of IPTPS*, 2007.

[8] Y. Huang, T. Z. J. Fu, D.M. Chiu, J. C. S. Lui, and C. Huang, “Challenges, design and analysis of a large-scale p2p-vod system,” in *Proceedings of the ACM SIGCOMM Conference on Data Communication (SIGCOMM ’08)*, pp. 375–388, August 2008.

[9] K. A. Hua, S. Sheu, and J. Z. Wang “Earthworm: a network memory management technique for large-scale distributed multimedia applications,” in *Proc.*

IEEE INFOCOM '97, vol. 3, Kobe, Japan, Apr. 1997, pp. 990–997.

[10] S. Sheu, K. A. Hua, and W. Tavanapong "Chaining: a generalized batching technique for video-on-demand systems," in *Proc. IEEE Int. Conf. Multimedia Computing and Systems*, Ottawa, ON, Canada, 1997, pp. 110–117.

[11] J. K. Chen and J. L. C. Wu "Adaptive chaining scheme for distributed VOD applications," *IEEE Trans. Broadcast.*, vol. 45, no. 2, pp. 215–224, Jun. 1999.

[12] Te-Chou Su, Shih-Yu Huang, Chen-Lung Chan, and Jia-Shung Wang "Optimal Chaining Scheme for Video-on-Demand Applications on Collaborative Networks," IEEE *RANSACTIONS ON MULTIMEDIA*, VOL. 7, NO. 5, pp-972-980, OCTOBER 2005.

[13] Z. Liu, Y. Shen, S. Panwar, K. W. Ross, and Y. Wang, "Efficient substream encoding for P2P video," *In 16th Packet Video Workshop*, Lausanne, 2007.

[14] D. Wu, Y. Liu, and K. W. Ross, "Queuing network models for multi-channel P2P live streaming systems," in *Proceedings of the IEEE 28th Conference on Computer Communications (INFOCOM '09)*, pp. 73–81, April 2009.

[15] M. Wang, L. Xu, and B. Ramamurthy, "Linear programming models for multi-channel P2P streaming systems," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM '10)*, 2010.

[16] C. Zhao, X. Lin, and C. Wu, "The streaming capacity of sparsely-connected P2P systems with distributed control," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM '11)*, pp. 1449–1457, 2011.

[17] S. Sengupta, S. Liu, M. Chen, M. Chiang, J. Li, and P. A. Chou, "Peer-to-peer streaming capacity," *IEEE Transactions on Information Theory*, vol. 57, no. 8, pp. 5072–5087, 2011.

[18] S. Liu, M. Chen, S. Sengupta, M. Chiang, J. Li, and P. A. Chou, "P2P streaming capacity under node degree bound," in *Proceedings of the 30th IEEE International Conference on Distributed Computing Systems (ICDCS '10)*, pp. 587–598, June 2010.