2008 14th IEEE International Conference on Parallel and Distributed Systems

Performance analysis of peer-to-peer video streaming systems with tree and forest topology

Paolo Giacomazzi, and Alessandro Poli
Department of Electronics and Information
Politecnico di Milano
giacomaz@elet.polimi.it, poli@elet.polimi.it

Abstract

Peer-to-peer networks are an increasingly popular solution for the distribution of media content to a large number of users, with limited investments for network infrastructures. The distribution of a real time video stream imposes strict performance requirements such as small playback delays and few frame losses. In this paper, we focus on peer-to-peer video streaming systems with tree or forest content distribution structure and we provide a sensitivity analysis to investigate the impact of three critical parameters – rejoin time, average permanence time of peers and playback threshold – over the quality of the video stream received by users. The study, carried out through simulation, considers a general peer-to-peer video streaming reference model with tree/forest topology.

1. Introduction

Peer-to-peer video streaming systems are used to distribute live video content among large sets of users. The distributor is not required to have large network infrastructures, since the distribution of content relies mainly on users' resources. Users, referred to as *peers*, receive the video stream from other peers and forward it to one or multiple peers. This multicast-like paradigm can be achieved by creating an overlay network over which the content is exchanged. This overlay network could have various topologies, such as a tree, a forest – i.e. multiple trees – or a mesh. In the literature, examples of systems using a single tree topology are Narada [1], Scattercast [2], and Zig-zag [3]; systems with a forest distribution topology are VidTorrent [4], Coopnet [5, 6], Nice [7], Overcast [8], and SplitStream [9]; systems with a mesh approach are CoolStreaming [10], GridMedia [11] and PPLive [12].

The analysis of the performance of peer-to-peer video streaming systems in the literature is carried out by means of three techniques: (1) *trace analysis* of working systems or the deployment and study of prototypal systems on specific *test beds*, (2) *analytical studies*, and (3) *simulation*.

The largest part of studies belongs to category (1). [13] presents a measurement study of the popular PPLive system [12], performed by means of a dedicated crawler. The cited work studied users' behaviour, peers' data exchange and playback delays. The authors of [14], using a combination of analysis and real traces of CoolStreaming [10], study how buffering techniques are used to cope with system dynamics and heterogeneity. Magellan [15] is a project launched with the objective of gaining in-depth insights on P2P streaming characteristics. The authors used 120 GB of traces from a commercial system to explore the behavior of some topological properties over the time. [16] provides a survey and a set of experiments on popular P2P video streaming systems, measuring performance indexes, such as the ratio of lost frames and the playback delay. Authors in [17] developed a framework used to analyze two popular P2P video streaming systems; they studied resource usage, locality, and stability of data distribution. In [18], an analysis platform for P2P video streaming is presented, taking into account the interactions between peers and the underlying network; the platform allows to connect a real P2P video client (purposely modified) to a network simulator, and to study the rate of loss frames for each peer.

[19] belongs to category (2); authors provide a stochastic model, used to compare different downloading strategies based on two performance metrics: probability of continuous playback, and startup latency; [20] proposes an analytical model of a real-time P2P video streaming system to estimate some performance parameters, such as the mean delay, and

1521-9097/08 $25.00 © 2008 IEEE
DOI 10.1109/ICPADS.2008.12

service provider revenue, as function of the nodes preemption probabilities.

Simulation (3) is used in [21], for a comparative study between the tree-based and the mesh-based approaches. The authors studied the effects of connections bandwidth, peer degree, bandwidth heterogeneity, group size, and churn. [22] is another analysis, performed by simulation, of a reference P2P video streaming system with Multiple Description Coding; it provides an evaluation of video quality.

Most of existing studies focus on the analysis of full systems *as-is*, without investigating the impact on performances of changes in their operational parameters.

In this paper, we provide a sensitivity analysis of the system performance to three critical parameters so far not studied thoroughly in the literature: *rejoin time*, *average permanence time of peers* and *playback threshold*. We carry out our study by means of a fine-grained simulative modeling of the peer-to-peer video streaming system. To the best of our knowledge, our simulative model of the system is significantly more accurate than similar extant models.

2. The reference system model

In this paper, we focus on tree-based peer-to-peer video streaming systems. Our model is inspired to VidTorrent [4], a tree-based peer-to-peer video streaming system developed at the Massachussets Institute of Technology. However, our model is more general and it accounts for peer-to-peer video streaming systems with the following properties: (1) a unique content distribution source is responsible for the provisioning of the video stream to the whole system, (2) the structure of the distribution is a tree or a forest, (3) at the application level, in the overlay peer-to-peer network, the content is organized into chunks of video frames referred to as *segments*, (4) a single *frame* can be split into a fixed number (≥1) of *sub-frames* of variable length, (5) users can join and leave the peer-to-peer system dynamically, even during the distribution of a video.

In the following sections the model of the reference system is explained in detail.

2.1. The video stream

The video stream provided by the source is a sequence of m ordered frames. We identify a single *frame* f_i by its frame number $i = 1, 2, \ldots, m$. For each frame we know the start time $f_i.start$ and the end time $f_i.end$ in seconds, identifying the time interval covered by the frame with respect to the entire video stream. Every frame is split into n *sub-frames*, where a sub-frame represents a part of the whole frame, such as, for example, a specific layer in a layered coding, or a single description in a Multiple Description Coding (MDC). A subframe sf_{ij} is identified by the frame number i and the sub-frame offset $j = 1, 2, \ldots, n$. When coping with single description/single layer codings, a frame comprises of one sub-frame ($n = 1$). For each sub-frame we know the length $sf_{ij}.length$ in bytes. Sub-frames can have either variable or fixed size.

2.2. Segments

At the application layer, chunks of k sub-frames are organized into segments. A segment s_i is assembled by grouping sub-frames having the same offset in consecutive frames (see Figure 1). For example, when $n = 4$ and $k = 3$, the first segment of the video stream s_1 is made of the sub-frames sf_{11}, sf_{21}, and sf_{31}, while s_2 is made of the sub-frames sf_{12}, sf_{22}, and sf_{32}.

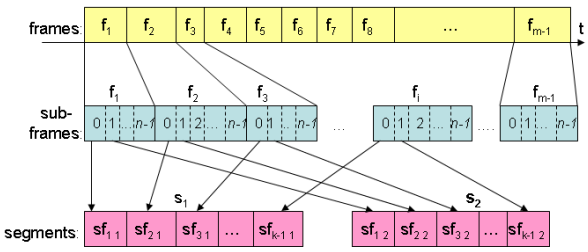

Figure 1. Frames, sub-frames and segments.

2.3. Source and trees

The video content is distributed among all peers through a set of q independent trees. The source provides the video stream and is placed at the root of each tree. It sequentially sends segments to its children at the rate determined by frame start and end times. The source has a limited amount of bandwidth Sup, measured in bit/s.

The number of trees q is a multiple of the number of sub-frames per frame n, such that only the segments composed by the sub-frames with the same j-th sub-frame offset are forwarded in the same tree. A variable number d of trees is allowed to transport segments with the same sub-frame offset (*tree diversity* property). The total amount of trees is thus $q = d \cdot n$.

Every segment is sent through the appropriate tree, alternating among the d available trees.

2.4. Trees and peers

A client in the peer-to-peer video streaming system is called *peer*. A peer, identified by p_i, in order to receive the video content, must be a node of the trees carrying the content. A peer is not required to be part of all trees. For example, it could be a node of the d trees transporting the segments made up of sub-frames with the same single sub-frame offset. With a Multiple Description Coding this would translate into the reception of a single description, causing the display of a degraded version of the video stream.

All peers, for each tree they are part of, receive segments from their parents and then send them to their children. A peer can be placed in different positions in different trees, and different trees can have a different topology.

2.5. Peers' operations

Each peer is responsible for the distribution of the media. It operates both at the overlay (application) and the underlay (network) levels. The access bandwidth of peer p_i, expressed in bit/s, is referred to as $p_i.Cup$ and $p_i.Cdown$, representing the upload and the download capacity of the access network, respectively. The peer is provided with a transmission buffer (in the upload direction) and a reception buffer (in the download direction).

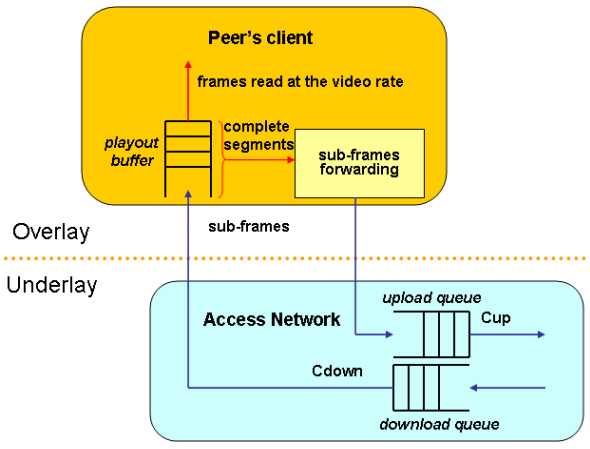

Figure 2. Peer's client and access network.

Each peer performs the following actions (see Figure 2):

1. The download queue, where the packets from parent peers are received, is emptied at a rate determined by $p_i.Cdown$.
2. All sub-frames received from the download queue are stored in an overlay buffer, named playout buffer, that reassembles the stream, according to frame numbers and sub-frame offsets.
3. As soon as all the sub-frames forming a segment are received, the segment is divided into packets and sent multiple times to all the children peers, through the appropriate trees. These packets are transmitted through the upload link.
4. The frames stored in the playout buffer are sequentially extracted by the client's player at the rate determined by the video stream.

2.6. Playout buffer

The playout buffer is responsible for the re-assembly of the video stream (segments are carried multiple packets in the underlay network). The playout buffer has a finite length, $PBlength$, measured in segments. When a sub-frame is received, it is placed in the correct position in the playout buffer. When a complete segment is reassembled in the playout buffer, it is sent to the children, as described in the previous section. When all the frames of a segment have been read by the player, a position in the playout buffer is freed.

A peer starts playing the video stream as soon as a playback threshold $PBTh$, measured in seconds, is reached. The reaching of the threshold is computed independently for every sub-frame offset. The playback starts as soon as, for at least one sub-frame offset (corresponding to a layer or a description in a layered or in a multiple description coding), the threshold has been exceeded.

2.7. Join and Leave

When a new peer wants to receive the video stream, it must become part of one or multiple distribution trees. In order to play the video stream, every peer must join at least the d trees transporting the segments composed of sub-frames with the same sub-frame offset.

The join procedure must be guided by the content of the sub-frames at different offsets. In particular, for peer p_i (see the example of Figure 3):

1. Depending on the free download bandwidth (equal to $p_i.Cdown$ if the peer is not already receiving any other sub-stream), the maximum number of sub-frames per frame to be received is computed.

2. The sub-frame offsets are chosen randomly.
3. For every chosen sub-frame offset, the peer selects a parent in all the d trees for that offset (a parent can be either another peer or the source). For each tree, the parent is randomly chosen among the peers at the highest level in the tree (nearer to the source) with a sufficient amount of free upload bandwidth.
4. After a time interval t_{join}, measured in seconds, the new node starts receiving the sub-streams from its new parents. This interval models the time required for the identification and the selection of a new peer.

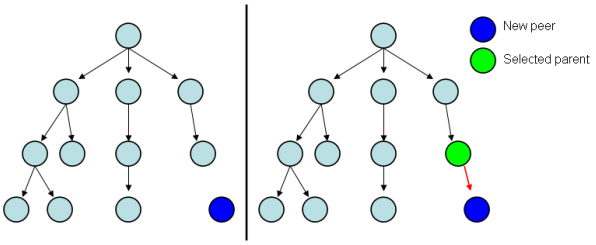

Figure 3. Join of a new peer.

A peer can leave the system unpredictably and without notification. Whenever this happens, its children – and, iteratively, all their grandchildren in the same tree – cease to receive the sub-stream. After a time interval t_{rejoin}, measured in seconds, the orphan peers start the join procedure for each sub-stream they are no longer receiving. This time interval can represent, for example, the time required by a keep-alive failure detection mechanism for the identification of the leave of a parent. The join procedure for an orphan peer is identical to the procedure followed by a new peer described above. Only direct children of the dead peer try to rejoin the trees in new positions, while all the isolated trees move together with their ancestors.

3. Performance analysis

In this paper, we analyze a peer-to-peer video streaming system where peers dynamically join and leave. The system, in the steady state, has a number N of simultaneously active peers. The time spent by a client in the system is exponentially distributed with average duration equal to $1/\mu$ s. Joins are independent Poisson events with a total average rate of joins equal to Λ s^{-1}, such that $N = \Lambda/\mu$.

3.1. Analysis parameters and indexes

In this paper we study the impact of three critical system parameters on the performance of the peer-to-peer video streaming system described in Section 2.

The selected performance parameters are:

1. *rejoin time*, $t_{rejoin} + t_{join}$, required by a peer to rejoin a tree when its parent leaves the system;
2. *average permanence time of peers* in the system, $\frac{1}{\mu}$, measured from peer's first join to its leave;
3. *playback threshold*, $PBTh$, that must be reached in peer's playout buffer before the video stream is locally played.

The first parameter strictly depends on the coordination protocol implemented by the particular peer-to-peer video streaming system. The average time spent by a peer in the system depends mainly on users' behaviour. The third parameter is a characteristic of the peer-to-peer video streaming client. Other system parameters may be equally important. In this work we have concentrated our attention on system features with a significant impact on performance, but not covered in depth by extant literature.

We measure system performance by means of the following indexes:

1. *playback delay*, defined as the time elapsing from the instant in which the source provides the content to the instant in which a client reads it from the peers' playout buffer;
2. *received frames and sub-frames ratios* for each peer, measured by considering the presence or absence of sub-frames in the playout buffer at their playback time.

Additional information is also registered, such as the number of total joins and leaves, and the topological characteristics of the distribution trees.

3.2. Simulation parameters

The system is analyzed through a simulation tool expressly implemented for this purpose. We have carried out an extensive simulation campaign by considering a peer-to-peer video streaming system with an average number of $N = 200$ active peers. We have fed the system with a real video trace of a soccer match with a duration of 40 minutes, coded with a Multiple Description Coding (MDC) with 4 descriptions per frame ($n = 4$). The average rate of the video stream is 820.5 kbps, and every frame has a fixed duration of 40 ms. Each segment comprises $k = 20$ sub-frames. A single description is sent through one tree ($d = 1$), such that a total number of $q = 4$ trees/sub-streams are

used. We have selected the source's bandwidth *Sup* in such a way that it can provide up to 20 sub-streams (i.e. up to 5 complete streams) simultaneously. The playout buffer length has been set equal to 160 s ($PBlength = 800$). When not otherwise explicitly notified, the total *rejoin time* is set to 100 s, the *average permanence time of peers* is equal to 15 minutes, and we use a *playback threshold* of 4 s. Simulations have been carried out in two different scenarios, considering the following upload and download access bandwidth ($p_i.Cup$ and $p_i.Cdown$) for all peers:

1. *Symmetric*: download 7 Mbps – upload 7 Mbps;
2. *Asymmetric*: download 7 Mbps – upload 1 Mbps.

The capacity of the transmission/reception buffers in the access network is infinite, so that frame losses are not caused by buffer overflows.

3.3. Results

In this section, the performance indexes presented in Section 3.1 are used to compare the behavior of the system as the rejoin time, average peer duration and playback threshold parameters vary. The reported numerical values have been obtained with a grand average over a set of independent simulations using different seeds for random number generation. The number of simulations for each point is variable and it has been chosen in such a way that the 95%-confidence intervals are smaller than 10% of the average values.

3.3.1. Rejoin time. Figure 4 shows the average percentage of sub-frames received by peers versus rejoin time while Figure 5 plots the percentage of peers receiving at least 95% of frames with at least one description (one sub-frame per frame).

With a rejoin time greater than 10 s, the quality of the received stream decreases significantly as rejoin time grows (frames are not received during rejoins). Quality degradation is sharper in the scenario with asymmetric access bandwidth, where every peer can at most upload one full stream to its children. This generates deeper distribution trees which are necessarily more sensitive to the temporary disconnections caused by leaves. The 10 s threshold for the total rejoin time seems to be critical. A coordination protocol enabling peers to quickly detect the death of their parents would be extremely beneficial for the system's performance.

Conversely, in all simulations we have found that playback delay is not affected by variations of the total rejoin time.

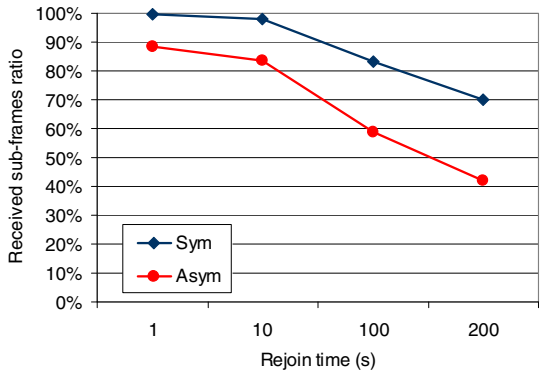

Figure 4. Ratio of received sub-frames with varying rejoin time.

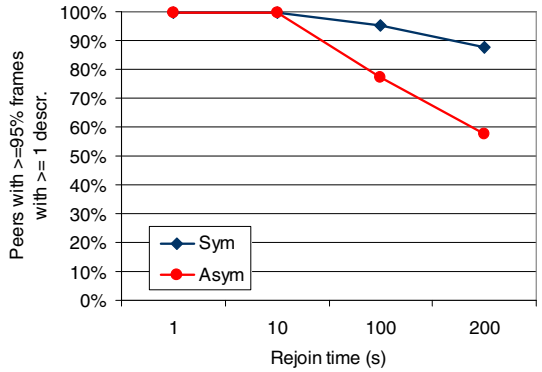

Figure 5. Peers with more than 95% of frames received with at least one description with varying rejoin time.

3.3.2. Average permanence time of peers. The average playback delay does not vary significantly as a function of the average permanence time of peers. However, as shown in Figure 6, the maximum playback delay noticeably increases when peers stay in the system for a short time. This behavior is even more stressed in the asymmetric scenario (the chart is not reported here). Remarkably, the average playback delay does not seem to vary as a function of the average permanence time of peers. However, the system can be very unfair, as in some cases a small number of peers can experiment a playback delay much higher than the average. For real-time events this feature is critical.

The quality of the received stream is alike influenced by the average permanence time of peers, as shown in Figure 7. When the system is less stable, i.e. peers frequently join and leave, frame losses increase and, consequently, the number of peers able to receive at least one description decreases.

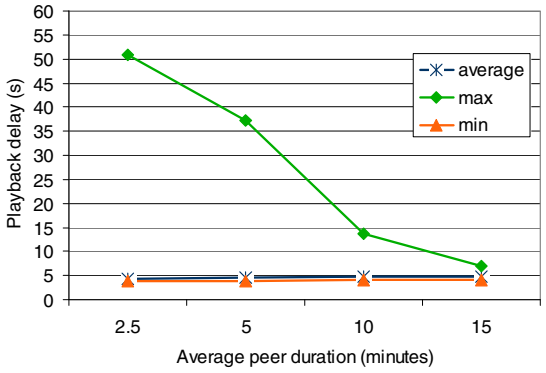

Figure 6. Average, maximum and minimum playback delay in the symmetric scenario with varying average permanence time of peers.

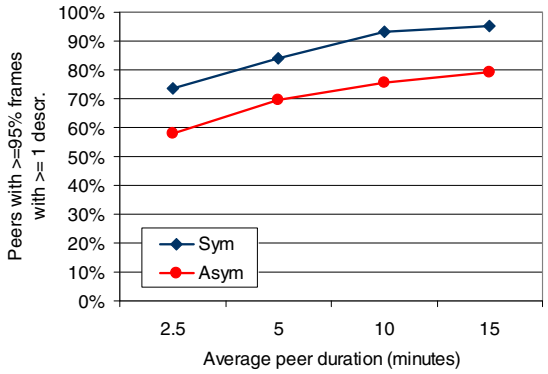

Figure 7. Peers with more than 95% of frames received with at least one description with varying average permanence time of peers.

3.3.3. Playback threshold. The playback threshold influences both playback delay and percentage of received frames. The larger the playback threshold, the longer is the delay experienced by peers. As shown in Figure 8, in the symmetric scenario, the playback delay is slightly above the playback threshold, while it is more than 3 s above the playback threshold in the asymmetric scenario. The effect on the user's quality of experience depends on the type of content and it is particularly critical with real-time events.

Conversely, the number of peers receiving at least one sub-frame (see Figure 9) reaches a maximum only when the playback threshold is above a given value, in this example above 4 s. Sub-frames losses are caused by an insufficient buffering of the video stream.

These results show the existence of a tradeoff between the delay experienced by peers and the quality of the distribution of the media.

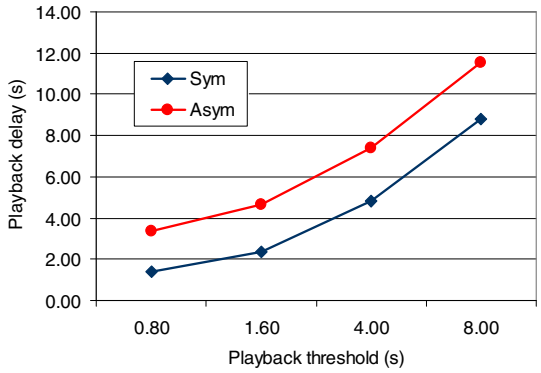

Figure 8. Average playback delay with varying playback threshold.

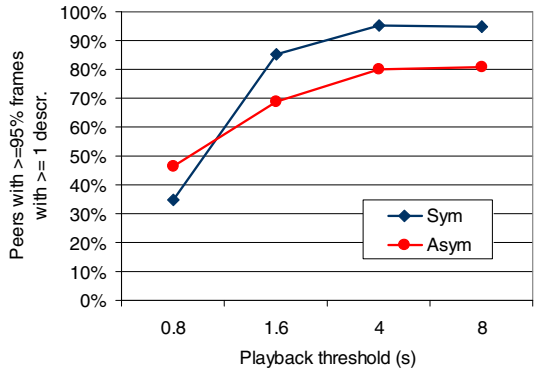

Figure 9. Peers with more than 95% of frames received with at least one description with varying playback threshold.

4. Conclusions

We have carried out through simulation a detailed sensitivity analysis of a peer-to-peer video streaming system with a forest topology. Our simulation model is based on the VidTorrent system, but it is significantly general and it can be applied to a large class of peer-to-peer video streaming systems, based on a tree or forest distribution structure. The simulation model is very detailed and it accounts for many important operations and features of the real system.

In our sensitivity analysis we have studied the impact of *total rejoin time*, *average permanence time of peers* and *playback threshold* on system performance.

Results show that total rejoin times of peers should not exceed a maximum value (in our scenarios around 10 s) above which the system's performance is greatly affected. In fact, the instability of the peer-to-peer video streaming system topology, caused by a limited amount of time spent by each peer in the system, is a key cause of the degradation of quality, in terms of playback delay and received frames. As far as playback delay is concerned, if the average permanence time of users is small (less than 15 minutes), the systems exhibits a sharply unfair behavior. In particular, the average user experiments a playback delay of around 5 s while the playback delay of the user receiving the worse treatment can be one order of magnitude greater.

Moreover, we have found that there exists a tradeoff between the playback delay and the ratio of received frames, driven by the amount of buffered video stream before starting the playback. Furthermore, the quality of the distribution of the media is also highly affected by the upload capacity of peers.

In this first work we have set up a simulation framework for the performance analysis of peer-to-peer video streaming systems with tree or forest topology. Our current research aims at devising and assessing the performance of user-rewarding schemes, in order to improve the system's performance by exploiting cleverly the resources of peers with higher transmission capacity and stability.

5. References

[1] Y.H. Chu, S.G. Rao, and H. Zhang, "A Case for End System Multicast", *Proc. ACM Sigmetrics Int'l. Conf. Meas. Modeling of Comp. Sys.*, 2000.

[2] Y.D. Chawathe, "Scattercast: An Architecture for Internet Broadcast Distribution as an Infrastructure Service", *Un. of California*, Berkeley, 2000.

[3] D.A. Tran, K.A. Hua, T.T. Do, "ZIGZAG: An Efficient Peer-to-Peer Scheme for Media Streaming", *22nd Annual Joint Conf. of the INFOCOM*, 2003.

[4] I. Mirkin, "Reliable Real-time Stream Distribution Using an Internet Multicast Overlay", http://viral.media.mit.edu/index.php?page=vidtorrent, 2006.

[5] V. N. Padmanabhan, K. Sripanidkulchai, "The Case for Cooperative Networking", *Proc. of the 1st Int'l Workshop on Peer-to-Peer Systems*, 2002.

[6] V.N. Padmanabhan, H.J. Wang, P.A. Chou, K. Sripanidkulchai, "Distributing Streaming Media Content Using Cooperative Networking", *ACM NOSSDAV*, 2002.

[7] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable Application Layer Multicast", *Proc. 2002 Conf. Apps., Technologies, Architectures, and Protocols for Comp. Commun.*, 2002.

[8] J. Jannotti, D.K. Gifford, K.L. Johnson, M.F. Kaashoek, J.W. O'Toole, "Overcast: Reliable multicasting with an overlay network", *Proc. OSDI*, 2000.

[9] M. Castro, P. Druschel, A.M. Kermarrec, A. Nandi, et al., "SplitStream: high-bandwidth multicast in cooperative environments", *Proc. of the 19th ACM symposium on Operating systems principles*, 2003.

[10] X. Zhang, J. Liu, B. Li, T. S. P. Yum, "CoolStreaming/DONet: A data-driven overlay network for peer-to-peer live media streaming", *IEEE Infocom*, 2005.

[11] L. Zhao, J.G. Luo, M. Zhang, W. J. Fu, J. Luo, Y.F. Zhang, et al., "Gridmedia: A Practical Peer-to-Peer Based Live Video Streaming System", *IEEE 7th Workshop on Multimedia Signal Processing*, 2005.

[12] PPLive, http://www.pplive.com/.

[13] X. Hei, C. Liang, J. Liang, Y. Liu, K.W. Ross, "A Measurement Study of a Large-Scale P2P IPTV System", *IEEE Transactions on Multimedia*, Dec. 2007, vol. 9(8), pp. 1672 – 1687.

[14] S. Xie, G.Y. Keung, B. Li, "A measurement of a large-scale peer-to-peer live video streaming system", *Packet Video 2007*, Nov. 2007, pp. 153 – 162.

[15] C. Wu, B. Li, S. Zhao, "Magellan: Charting Large-Scale Peer-to-Peer Live Streaming Topologies", *ICDCS '07, 27th International Conference on Distributed Computing Systems*, June 2007, pp. 62.

[16] A. Sentinelli, G. Marfia, M. Gerla, L. Kleinrock, S. Tewari, "Will IPTV ride the peer-to-peer stream? [Peer-to-Peer Multimedia Streaming]", *IEEE Communications Magazine*, June 2007, vol.45(6), pp. 86 – 92.

[17] S. Ali, A. Mathur, H. Zhang, "Measurement of commercial peer-to-peer live video streaming", *First Workshop on Recent Advances in Peer-to-Peer Streaming*, August 2006.

[18] M. Barbera, A.G. Busà, A. Lombardo, G. Schembra, "CLAPS: A Cross-Layer Analysis Platform for P2P Video Streaming", *ICC '07. IEEE International Conference on Communications*, June 2007, pp. 50 – 56.

[19] Y. Zhou, D.M. Chiu, J.C.S. Lui, "A Simple Model for Analyzing P2P Streaming Protocols", *ICNP 2007, IEEE International Conference on Network Protocols*, Oct. 2007, pp. 226 – 235.

[20] G. Incarbone, G. Schembra, "Peer admission control in a real-time P2P video distribution platform: definition and performance evaluation", *Packet Video 2007*, Nov. 2007, pp. 163 – 172.

[21] N. Magharei, R. Rejaie, and Y. Guo, "Mesh or Multiple-tree: A comparative study of live P2P streaming approaches", *Proc. of IEEE INFOCOM'07*, May 2007.

[22] I. Lee, Y. He, L. Guan, "Centralized P2P Streaming with MDC", *IEEE 7th Workshop on Multimedia Signal Processing*, Oct. 2005, pp. 1 – 4.