A Measurement Study of a Peer-to-Peer Video-on-Demand System

Bin Cheng[*], Xuezheng Liu[§], Zheng Zhang[§], Hai Jin[*]
[*]Services Computing Technology and System Lab, Cluster and Grid Computing Lab
Huazhong University of Science and Technology, Wuhan, China
[§]Microsoft Research Asia, Beijing, China
[*]{*showersky,hjin*}@hust.edu.cn [§]{*xueliu,zzhang*}@microsoft.com

ABSTRACT

Despite strong interest in P2P video-on-demand (VoD) services, existing studies are mostly based on simulation and focus on areas such as overlay topology. Little is known about the effectiveness of P2P in VoD systems and the end user experience. In this paper we present a comprehensive study of these issues using the two-month logs from a deployed experimental P2P VoD system over CERNET[1]. Our key findings are: (1) the key factor is the popularity of channels and a moderate number of concurrent users can derive satisfactory user experience. However, good network bandwidth at peers and adequate server provisioning are critical. (2) a simple prefetching algorithm can be effective to improve random seeks. Overall, we believe that it is feasible to provide a cost-effective P2P VoD service with acceptable user experience, and there is a fundamental tradeoff between good experience and system scalability.

1. INTRODUCTION

Unlike other popular P2P services such as file downloading [5][10] and live streaming [7][8][12], the more challenging P2P VoD (video-on-demand) service is relatively less understood. This paper reports our initial empirical study of GridCast, a P2P VoD system that has been deployed for half a year.

Our initial attempt is to define important metrics that will guide further improvements; the current limited scales allow us to perform extensive instrumentations to identify optimization opportunities on both server and client side. Along the way, we gained a few insights. For instance, even with limited concurrent users that are spread in various playback positions, the performance already approaches that of an ideal system. However, the benefit only extends to peers that enjoy good bandwidth and when server stress is less an issue. Peers with low bandwidth not only have poor user experience, but can also cause problems to other well-connected peers. We find that forward seek dominates backward seek with a 7:3 split, and that around 80% of seeks are within short distance. We also find that a simple prefetching algorithm is effective to improve random seeks, but further optimizations are necessary.

We give a brief description of GridCast in Section 2. Section 3 explains our experiment methodology and the collected dataset. Section 4 presents an in-depth analysis of the overall system performance and user experience. We discuss related work in Section 5 and conclude in Section 6.

2. SYSTEM OVERVIEW

Just like other P2P content distribution systems, GridCast uses a set of *source servers* to release media files to participating peers, who asynchronously play the files while exchanging data among themselves. Unlike file downloading and live streaming, a peer is more "selfish" in the sense that it only cares about contents after its current playing position, which is often different from other peers. Most of the time, a peer's downloading targets are those whose playback positions are ahead, and it can only help those that are behind. However, a peer can also change its playing position at any time. These characteristics make a VoD system harder to optimize, rendering globally optimal strategies such as "rarest first" as employed in BitTorrent[5] inapplicable.

To cope with the above problem, a GridCast peer maintains a routing table consisting of some peers that are placed in a set of concentric rings with power law radii, distanced using relative playback positions (Fig.1), and uses gossips to keep the routing table up-to-date. This architecture allows a peer to find a new group of position-close partners in logarithmic steps after it seeks to a new playing position. The *tracker* can

[1]China Education Network

This work was partially supported by NSFC grant No.60433040.

be considered as a stationary peer whose playback position stays fixed at time zero. The tracker's only job is to keep track of its membership view, which bootstraps any new peers.

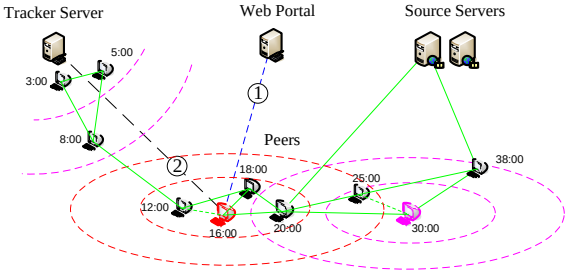

Fig. 1 Architecture overview

The peer caches played content onto its local disk. These data are served to other peers, or to itself in case of backward seeks. The media file is divided into chunks. Each chunk has one second play time. A peer exchanges chunks with its (up to 25) partners, starting from those in the innermost ring and then outwards, or from the source server otherwise. The peer fetches the next 10 seconds first; the playback stalls if these data are not fetched in time. Next, it tries to fetch the next 200 seconds. If bandwidth allows, the peer also tries to fetch *anchors*. As Fig. 2 shows, anchors are segments each consisting of 10 continuous seconds, and are distributed throughout the media file with fixed interval (e.g. 300 seconds). When a seek occurs, we adjust the playback position to the beginning of the closest anchor if the anchor has been already downloaded. Thus, the seeking is satisfied instantly and the playing time of that anchor overlaps with the time needed for the peer to establish partners at the new position. Anchor prefetching is a new addition to the system, and other details can be found in [2].

3. EXPERIMENT METHODOLOGY

3.1 Deployment

GridCast is deployed since May 2006 and has attracted more than twenty thousand users and supported up to hundreds of concurrent users at the peak time with one source server, which has 100Mb bandwidth, 2GB Memory and 1TB disk. During the popular 2006 FIFA WORLD CUP event, it provided VoD service for users in 6 provinces in China. A majority of users are students with Internet services supplied by CERNET. There are also users from other ISPs in China. The video programs typically have bitrate from 400 Kbps to 600 Kbps (with a few channels exceeding 800 Kbps), and are either Window Media Video (WMV) or Real Video (RMVB) file format. The contents are classified into 9 subsections, including *Sports*, *Science*, *Cinema*, *Leisure* and others. Each published media file is called a *channel*, and a client's playback activity is termed as a *session*.

Although the current scale is still limited, the mixed network environment and the detailed logs enable us to perform an in-depth analysis of a running P2P VoD system.

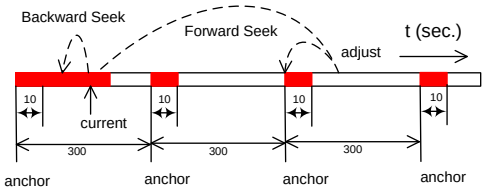

Fig. 2 Anchor distribution

3.2 Data Collected

A GridCast client is instrumented to collect logs that are turned on during playing time, and the logs are uploaded to the tracker periodically. These session logs keep the most important information on seek operations, buffer maps, jitter and anchor usage. We record the snapshots taken at all peers with a 30-second granularity. The source server keeps other statistics such as total bytes served. Table 1 gives an overview of the collected dataset.

Table 1 Log statistics

Log duration	~ two months
Number of visited users	~ 20,000
Percent of CERNET users	98%
Percent of non-CERNET users	Netcom: 1% Unicom: 0.6% Telecom: 0.4%
Pair-wise peer bandwidth	CERNET: >100KB/s Non-CERNET: 20~50KB/s CERNET to non-CERNET: 4~5KB/s
Percent of NAT users	22.8%
Maximal online users	~ 360
Number of sessions	~ 250,000
Number of videos	~ 1,200 channels
Average Code rate	500~600kbps
Movie length	on average about an hour
Total served from the source server	11,420GB
Total played by clients	15,083GB

To evaluate the effectiveness of the anchor prefetch algorithm, in Sep. and Oct. 2006, we ran two experiments, with and without the anchor prefetch, and each lasted one month. The switch is made by directing client code to different

codepaths when the user joins. As the CERNET users generally have good network bandwidth (>100KB/s), non-CERNET users are much lower and have asymmetric uplink/downlink bandwidths. The pipe between CERNET and the commercial ISPs is much narrower, at around 4~5KB/s.

4. EXPERIMENTAL RESULTS

4.1 Overall System Performance

In Fig. 3, the top curve represents the total contents played during a typical day, normalized to the peak which occurred around 1:00PM. This curve, labeled *cs*, represents the stress of a traditional client/server model where all data are streamed from the source. The bottom curve, labeled *ideal*, is the total amount of data served by the source if it serves only one user with the bit-rate of the file per channel. This is the minimum possible stress, bringing it down further would require the peers themselves to register as sources in the system. The middle curve is what our source server actually serves.

Unsurprisingly, there is a clear pattern that the stress follows users' daily activity schedule. The stress in hot time (when user population is large, e.g., 12:00PM to 1:00PM) sees 3.2 and 4.6 times increases compared to that of the cold time (when population is small, e.g., 9:00AM to 10:30 AM), for ideal and cs, respectively. The fact that during hot hour(s) the number of active channels increases has a consequence to both server load and user experience, as we will explain later.

The stress at the source follows closely to the ideal curve, demonstrating that the system works fairly effectively at this scale. The gap between the two curves is due to a number of factors: the source needs to provide the data anytime when the peer can not obtain data from other peers. This happens more frequently when the user joins the system or seeks a new playing position, or when its partners don't have enough data or bandwidth. To understand the issues further, we define the *utilization* of a channel to be the ratio of data served from peers to the total fetched data. The higher the utilization, the better the peers are helping each other. In the ideal model, the utilization of a channel with n concurrent users is $(n\text{-}1)/n$ because only the foremost peer is required to fetch data from the source server.

Fig. 4 plots the utilization against channel popularity (i.e. concurrent users). Evidently, the utilization quickly approaches the ideal one with more peers. Fig. 5 examines in finer details why the gap exists. Our log includes the snapshots of the availability of downloaded chunks of all joined peers and the amount of data retrieved from the source and from local partners every 30 seconds. A peer will retrieve from the server if 1) the content exists only in the source server, 2) the content does not exist in its local partners but exists in disconnected peers because of NAT and 3) its partners do not have sufficient bandwidth to meet the demand. All of these problems will contribute to lower utilization, but they become progressively less severe with more peers. With limited peers, missing content (most likely due to seek operations) and NAT are the two primary reasons. Interestingly, there are points where the utilization is even better than the ideal case. This is mostly due to paused peers or finished peers that still stay online, which act as a temporary source server.

4.2 User Experience

Good user experience is critical for VoD services. The metrics that we examine are startup latency, seek latency, and jitter.

A. Startup and Seek Latency

After a peer joins a channel, it starts to play only after receiving 10 seconds worth of content. The same goes for seeks. These data come from the

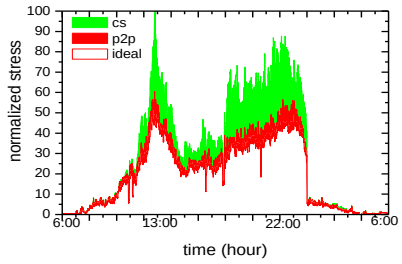

Fig. 3 Bandwidth consumption over time during a typical day

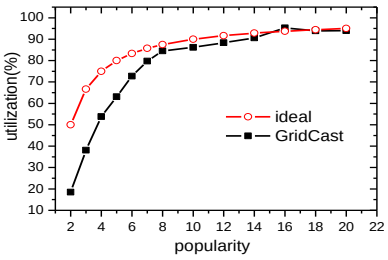

Fig. 4 Utilization vs. popularity

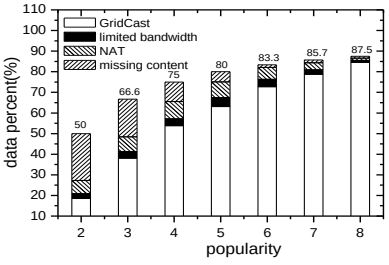

Fig. 5 Reasons for the gap between GridCast and the ideal model

source server if the peer is the only one in the channel, or up to ten most content-close peers as recommended by the tracker. Like an ordinary peer, the tracker also maintains a list of peers on a set of concentric rings. These two latencies are qualitatively the same since the user jumps to certain playing point.

Fig. 6 provides the cumulative distribution functions (CDF) of startup latency and seek latency of all sessions. Although the startup latency has a wide distribution up to 60 seconds, more than 70% and 90% of sessions have lower than 5 and 10 seconds, respectively. Seek latency is smaller, more than 70% and 90% of the sessions have lower than 3.5 and 8 seconds, respectively. There are two reasons why seek latencies are better. First, startup latency encounters a 2-second connection delay as the user establishes the initial partners if there are any (and that's the reason the two are equal when there are no partners). Second, as we will explain in Section 4.3, many seeks are forward and short seeks. These seeks would find data readily prefeched from its partners.

These results are not surprising given that a majority of the users are within CERNET, up to 98%. We break down the startup and seek latency according to whether the user is in the same campus, not in the same campus but still on CERNET, and those that are from elsewhere (Fig.7). In contrast to other CERNET users, the campus users enjoy shorter latency due to their better network locality. However, the non-CERNET users encounter poor latency because they have lower network capacity (4~5KB/s). These data suggests that if the non-CERNET users increase in the future, optimizing their latency is critical since those users are seeing a delay close to 1 minute.

More peers will definitely help, as the needed contents can be downloaded in parallel (Fig. 8). Both curves drop quickly with more peers, with startup latency elevated by the 2-second initial connection delay. Ideally they should drop in reverse proportion to number of peers. However, this is not true especially for startup latency, and the difference lies in the heterogeneity in peers' networking capability. This can be seen more clearly in Fig. 9, which is a scatter plot of startup latencies of the on-campus users. While the startup latency generally drops with more initial peers, there are always outliers that suffer from longer startup latencies. A close look at these data points reveals that, inevitably, there are non-CERNET users in the initial peers. Our current algorithm is such that the initial content is evenly partitioned and downloaded from those peers, and the slowest one determines the startup latency. A future improvement will be to be more network-aware when selecting the initial peers.

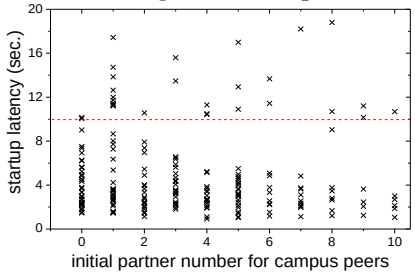

Fig. 9 Startup latency distribution of campus users

B. Jitter

We use the number of delayed chunks and their percentage to analyze the jitter of playback. Similar to the latency of seek and startup, jitter is related to the network capacity of peers. The non-CERNET users always encounter serious jitters because their available bandwidths with the source server are too limited (4~5KB/s) and there are not enough non-CERNET partners to help them. CERNET users do far better, as shown in Fig. 10. There are 72.3% and 40.6% sessions without any jitters for 5 minutes and 40 minutes durations, respectively. The sessions with jitter

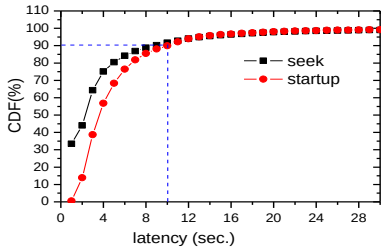

Fig. 6 CDF of startup latency for all sessions

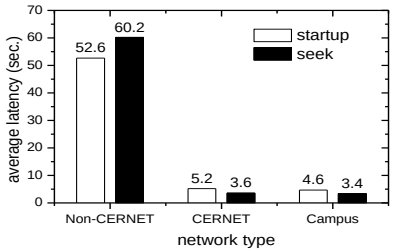

Fig. 7 Startup latency for users in different networks

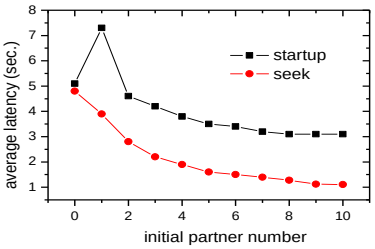

Fig. 8 Average latency vs. partner number

encounter 3~4% delayed data for sessions that last more than 20 minutes. Again, GridCast works fairly well for users with good network connectivity.

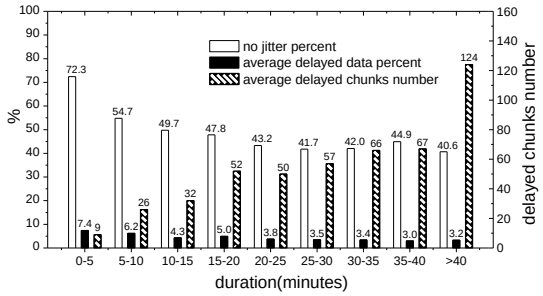

Fig. 10 Jitters vs. duration (CERNET users)

However, even CERNET users can have poor user experiences. Fig. 11 presents the normalized unacceptable seek (defined as latency >10 seconds) and jitter (defined as longer than 100 seconds duration) of an entire week, aligned on hours. It is obvious that the fluctuations have a strong correlation with the stress at the source server. In GridCast as well as in a number of proposed systems, the source is to provide data whenever a peer is unable to obtain them from its partners.

Server stress peeks when there are more users in the system. But the amount of users is not the reason for server stress increase. During the hot time, the number of active channels is about 3 times of that in cold time, and this causes high server stress that results in poor user experience. In Fig. 12, we see that in cold time both hot channels and cold channels deliver good user experience, with only 3.5% unacceptable jitter and 2.5% unacceptable seek, respectively. This is in sharp contrast to the hot time. However, user experience is heavily influenced by content popularity. With more concurrent users, each peer relies less on the source server but more upon each other, thus their experience steadily improves.

One solution to reduce server load is to let peers to register as sources in the system. However, it is not necessarily the only long term solution. We believe that to deal with unexpected demand fluctuations, a hybrid VoD system such as GridCast must have proper QoS policies implemented on the server side. This can range from delisting to giving lower priority to the less popular contents during hot hours.

4.3. Optimization

We start this section by analyzing seeking behaviors that we have observed (Table 2). There is a 7:3 split of forward versus backward seek. Furthermore, short distance seeks dominate: about 80% of seeks in either direction are within 300 seconds. This suggests that prefetching the next anchor relative to the current play position can be effective. The seeking behavior also has a strong correlation with the popularity. As Fig. 13 shows, there are fewer seeks in more popular contents, which generally also have longer sessions.

Table 2 Statistics of seek operations

Forward	72%	Short (<300 sec.)	81%
		Long (>300 sec.)	19%
Backward	28%	Short (<300 sec.)	76%
		Long (>300 sec.)	24%

As mentioned earlier, we have implemented an anchor prefetching mechanism as a way to deal with random seeks. Fig. 14 compares the distribution of seek latency with and without anchors, collected from two different experiments. It is obvious that the anchor prefetching is effective. The fact that backward seeks see less benefit is not surprising because there are less backward seeks to begin with, resulting in less "holes" behind the current playing position. In the forward directions, the number of seeks that finish within one second jumps from 33% to 63%. Consider that about 80% of seeks are within 300 seconds, there is obviously more room to improve. Prefeching the next anchor is statistically optimal

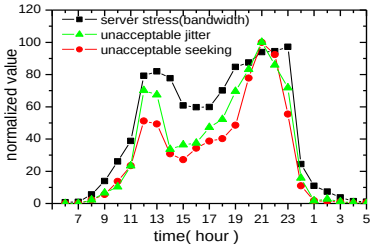

Fig. 11 Unacceptable jitter and seeking latency vs. server stress

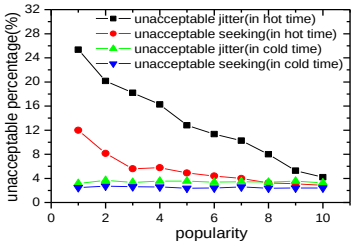

Fig. 12 User experience comparison at hot time and cold time

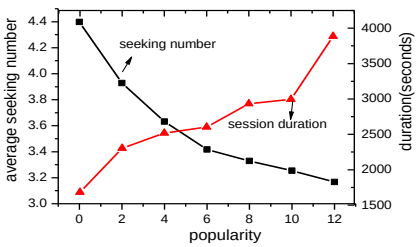

Fig. 13 Average seeking number & duration vs. popularity

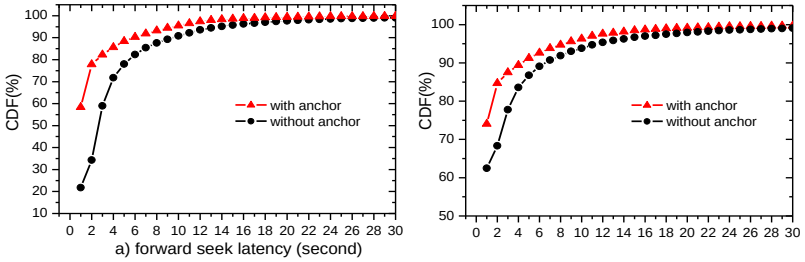

Fig. 14 Seeking latency comparison between with anchor and without anchor

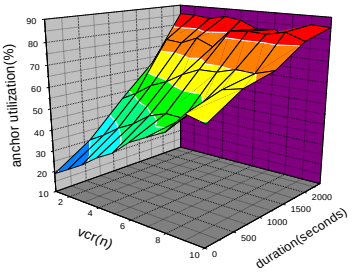

Fig. 15 Anchor utilization

from the individual user's point of view. On the other hand, "rarest-first" as currently employed is globally optimal in reducing the source server's load as it releases the rarest content from the source into the peers. Thus, the best strategy needs to consider both to reach the optimal tradeoff. Furthermore, past sessions can provide guidance: the parts that were played more are obvious candidates for prefetching, as proposed in [11]. These options will be evaluated for the next release of GridCast.

The prefetching mechanism must be cost-effective, meaning that in addition to the reduction of seek latency, we need to understand their utilization. The metric *anchor utilization* is the ratio of the amount of played versus fetched anchors, shown in Fig. 15. Utilization rises with session duration as well as more seek operations. For all sessions, the utilization averages to 70%, we believe further optimization is still possible to improve the efficiency.

5. RELATED WORK

Most of existing work about P2P VoD [3][4][6] systems was concentrated on the protocol design of topology and the analysis of simulation results, including our previous work [2]. Different from them, our study provides real world results that can be used to validate simulation scenarios. Recently, Yu et al [9] presented an in-depth understanding of access patterns and user behaviors in a centralized VoD system. Zheng et al [11] proposed a distributed prefetching scheme for random seek support in P2P streaming application through the analysis of user behaviors log obtained from a traditional VoD system. Compared with them, our study provides insights for user experience and overall performance systems through the analysis of the trace obtained from a real P2P VoD system.

6. CONCLUSIONS

In this paper we presented a measurement study of a P2P VoD system deployed over CERNET, called GridCast. Our study demonstrates that peer-to-peer is capable of providing a cost-effective VoD service with acceptable user experience, even with moderate number of cooperative peers. We also found that simple prefetching algorithm can greatly reduce seek latency. We also identified a number of problems. For instance, more concurrent users can drive up number of active channels, leading to server stress growth and degrading user experience for peers with fewer partners. Also, peers with poor network connectivity are not well supported. These insights are helpful to improve future design of P2P VoD systems.

REFERENCES

[1] http://grid.hust.edu.cn/gridcast

[2] B. Cheng, H. Jin, and X.F. Liao, "RINDY: A Ring Based Overlay Network for Peer-to-Peer on-Demand Streaming", *Proceedings of Ubiquitous Intelligence and Computing*, Wuhan, 2006.

[3] T. Do, K. A. Hua, and M. Tantaoui, "P2VoD: providing fault tolerant video-on-demand streaming in peer-to-peer environment", *ICC'04*, 2004.

[4] Y. Guo, K. Suh, J. Kurose, and D. Towsley, "A peer-to-peer on-demand streaming service and its performance evaluation", *Proceedings of ICME'03*, Jul. 2003.

[5] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang, "Measurements, Analysis, and Modeling of BitTorrent-like Systems", *Proceedings of ACM IMC'2005*, Berkeley, CA, USA, Oct. 2005.

[6] C.S. Liao, W.H. Sun, C.T. King, and H.C. Hsiao, "OBN: Peering for Finding Suppliers in P2P On-demand Streaming Systems", *Proceedings of the Twelfth International Conference on Parallel and Distributed Systems (ICPADS'06)*, Jul. 2006.

[7] X. F. Liao, H. Jin, Y. H. Liu, Lionel M. Ni, and D. F. Deng, "AnySee: Peer-to-Peer Live Streaming", *Proceedings of INFOCOM'06*,. 2006.

[8] N. Magharei, R. Rejaie, "Understanding Mesh-based Peer-to-Peer Streaming", *Proceedings of NOSSDAV'06*, Rhode Island, May 2006.

[9] H.L. Yu, D.D. Zheng, B. Y. Zhao, W.M. Zheng. "Understanding User Behavior in Large-Scale Video-on-Demand Systems", *Proceedings of Eurosys'06*, Belgium, 2006.

[10] M. Yang, Z. Zhang, X.M. Li, and Y.F. Dai, "An Empirical Study of Free-Riding Behavior in the Maze P2P File-Sharing System", *Proceedings of IPTPS'05, New York*, USA, Feb. 2005.

[11] C.X. Zheng, G.B. Shen, S.P. Li, "Distributed prefetching scheme for random seek support in peer-to-peer streaming applications", *Proceedings of the ACM workshop on Advances in peer-to-peer multimedia streaming*, Hilton, Singapore, 2005.

[12] X.Y. Zhang, J. Liu, B. Li, and T.S. P. Yum, "CoolStreaming/DONet: a data-driven overlay network for peer-to-peer live media streaming", *Proceedings of IEEE INFOCOM'05*, Mar. 2005.