Topology dynamics in a P2PTV network

S. Tang, Y. Lu, J. Martín Hernández, F.A. Kuipers, and P. Van Mieghem

Delft University of Technology, P.O. Box 5031, 2600 GA Delft, The Netherlands
{S.Tang, Y.Lu, J.MartinHernandez, F.A.Kuipers, P.F.A.VanMieghem}@tudelft.nl

Abstract. In recent years, a number of commercial peer-to-peer TV (P2PTV) applications have been launched. Yet, their mechanisms and characteristics are unknown. In this paper, we study SopCast, a typical proprietary P2PTV system. Treating SopCast as a black box, we perform a set of experiments that are suitable to analyze SopCast in depth. We attempt to disclose the SopCast protocol. The dynamic nature of the SopCast overlay, in terms of node degree, is also addressed in this paper. Our approaches in analyzing the SopCast mechanism and characterizing its topological properties reveal important design insights in SopCast, and may help to better understand similar P2PTV systems.
Keywords: Topology dynamics, SopCast, P2PTV

1 Introduction

The success of peer-to-peer (P2P) file-sharing systems has spurred the deployment of P2P technologies in many other bandwidth-intensive large-scale applications. Peer-to-Peer Television (P2PTV) has become a popular means of streaming audio and video content over the Internet. Example applications are CoolStreaming [8], TVAnts[1], TVU[2], SopCast[3], etc. It is important to evaluate the traffic impact of such applications, while modeling their behavior. However, P2PTV streaming systems, such as SopCast, are developed for commercial purposes: thus, very little is known about their architectures. Some papers claim that SopCast is based on similar principles as those underlying CoolStreaming, e.g. [2], some refer to it as a BitTorrent-based P2PTV system, e.g. [1], but all without substantiating their claims. Furthermore, SopCast traffic is encoded, which makes understanding the protocol even more challenging. In this paper, we will investigate the SopCast P2PTV system by answering the following two questions. *What is the operational mechanism in SopCast? How are topology dynamics reflected in the SopCast overlay?*

The rest of the paper is organized as follows. In Section 2, related work is discussed. Section 3 describes the measurement settings in PlanetLab[4] network. Based on the experiments conducted in Section 3, we present our understanding

[1] http://tvants.en.softonic.com/
[2] http://www.tvunetworks.com/
[3] http://www.sopcast.org/
[4] http://www.planet-lab.org/

of the SopCast protocol in Section 4. Our methodology of modeling the time-variant SopCast overlay and the derived results are provided in Section 5. In Section 6, we conclude the paper.

2 Related measurement studies performed with SopCast

Ali *et al.* [6] evaluated the performance of both PPLive[5] and SopCast. They conducted experiments on a single host joining a system. The systems were run under different environments and the collected data was further analyzed to give insight into SopCast's operation. Silverston *et al.* [7] analyzed the different traffic patterns and underlying mechanisms of several P2PTV applications. Their study is based on a single measurement day. The dataset collected in this paper is from two personal computers that are directly connected to the Internet from their campus network. Sentinelli *et al.* [1] performed two sets of experiments on SopCast. There were in total 27 peers connected to a popular SopCast channel in the first experiment, while the second experiment was run on the PlanetLab network with 1 node acting as the source and 10 nodes performing as peers.

Most of the previous work is executed from a single point of observation [7], or from a small number of vantage points [1], [6]. Thus, results from these papers cannot accurately reflect the performance of the entire SopCast network. Furthermore, characterizing dynamic peer-to-peer networks has been considered a research field with a lot of contentions. The major dispute is on the accuracy of the captured topology snapshots in reflecting the actual peer-to-peer overlay. Stutzbach *et al.* [3] quantified the effects of the crawling duration and the ratio of unreachable peers on deriving network characterizations.

In this paper we study the SopCast protocol and provide our solutions to evaluating the topological properties (with respect to the degree distribution) and dynamics in a P2PTV network.

3 Experimental settings

We have used PlanetLab for our experiments, because it allows us to evaluate the entire constructed overlay. The experiment consists of two types of nodes.

A standard personal computer located in our campus network, which acts as the *source provider*[6] (SP). With the SP, we registered a dedicated channel to the SopCast network. In this channel, a small cartoon movie with a duration of 2 minutes and size of 3.5 MBytes is continuously broadcast in a loop. Thus, our experiment resembles a streaming system. The SP runs Windows XP. It is equipped with an Intel Pentium 2.4 GHz processor, 512 MB RAM and a 10/100 FastEthernet network interface, which is further connected through a router to the Internet. The second type of nodes are PlanetLab nodes that act as

[5] http://www.pplive.com/

[6] The source provider is the node who broadcasts the entire video by using SopCast software.

SopCast peers viewing the TV channel released by us. Each of the 51 PlanetLab nodes under consideration runs the following software: (1) SopCast Client (Linux version), with command line control; (2) Tcpdump[7] to enable passive monitoring on the traffic transmitted at the SopCast peers; and (3) Perl[8] Scripts to remotely control the PlanetLab nodes.

We conducted a single experiment on 11:00 am, August 22^{nd}, 2008. The experiment lasted for roughly 40 minutes. The 51 PlanetLab nodes were controlled in such a way that they joined and left the network simultaneously. We collected the traffic log files captured by tcpdump from all the 51 peers. Traffic collection at the SP is accomplished with Ethereal [4]. The collected trace files from the 51 PlanetLab nodes are further processed and analyzed with AWK[9] scripts.

4 Dissecting the SopCast protocol

SopCast is a proprietary P2PTV application and consequently the SopCast website only provides limited information about SopCast's video delivery mechanism. Although the work presented in this paper has been conducted in a thorough and careful way, without having the source code of SopCast, the claims that we have made are based on our investigation of SopCast. They are not the exact description of the protocol. In this paper, we only present our conclusions on the peer communication scheme and video delivery rule in Sopcast, because they play important roles in determining the topological properties in SopCast.

4.1 Identification of SopCast packets

Tcpdump reveals that SopCast relies on UDP. Since we are not able to decode the SopCast traffic, it is not possible to tell exactly what kind of messages are being exchanged in the captured trace files. To figure out the packet functionalities, we studied the packet lengths and the corresponding delivery patterns. Table 1 presents our findings.

Table 1. Summary of SopCast traffic

Type	Size (bytes)	Functionality
Video packet	1320	Maximum size of the video packets
	377, 497, 617, 1081, 1201	Video fragments
Control packet	52	HELLO packet to initiate link connections
	80	Confirmation on receiving the HELLO packet
	28	Acknowledgement
	42	Keep-alive message with neighbors
	46	Video requesting packet

[7] http://www.tcpdump.org/
[8] http://www.perl.org/
[9] AWK is a general programming language that is designed for processing text-based data, either in files or data streams.

4.2 Neighbor communication in SopCast

Assuming that a SopCast peer has retrieved a random list of peers (a *peerlist*) in the network, it will start to choose some peers with whom connections are established. If two peers exchange a 42-byte control packets with each other, we refer to these peers as being *neighbors*.

Communication between two peers in SopCast is always initiated by a $52-80$ byte packets pair. Once the connection is established, the pair of peers keeps exchanging a sequence of 42-byte packets with each other. The 42-byte packets are transmitted with a high frequency, roughly every second. We denote this packet as a *keep-alive* packet. With the keep-alive packets, a peer announces its existence in the network. The purpose is to maintain neighbor relation with others via the decentralized communication between peers. Peers can lose neighbors. In case a neighbor does not respond to the keep-alive packets, the peer stops contacting this neighbor until it chooses the neighbor again. A graphic illustration of the neighbor communication scheme is shown in Fig. 1(a).

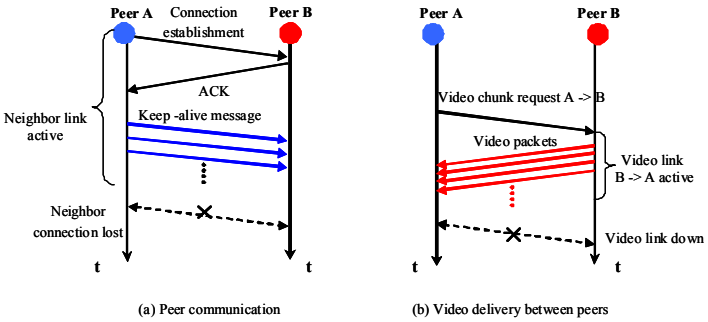

Fig. 1. Diagram of (a) neighbor communication and (b) video delivery in SopCast.

4.3 Video delivery in SopCast

Video delivery in SopCast is chunk-based. The TV content in SopCast is divided into *video chunks* or *blocks* with equal sizes of 10 kbyte. This finding is in line with the video chopping algorithm implemented in many existing P2P systems, although the chunk size may be different[10]. If a peer is providing video packets to its neighbors, we refer to the peer as a *parent*. A *child* is defined if a peer is receiving video packets. A peer is allowed to have multiple parents and multiple children. A parent-child relation can be established only when two peers are neighbors. A peer is free to request multiple video blocks from its parents.

[10] In Bittorrent, the default size of the chopped block is 256 kbytes.

A peer in SopCast never voluntarily delivers video streams to its neighbors. To download video packets, a child always needs to request them from its parent(s) via a *video request packet* with the size of 46 bytes, see Fig. 1(b). After receiving the request, its parent(s) will deliver a series of video packets to the child. In case the child needs more blocks, it sends another request. The requested blocks are treated as a large datagram during transmission. Due to the IP fragmentation principle, a large datagram such as 10 kbytes should be segmented into smaller pieces in order to pass over the Ethernet. SopCast sets the maximum size of the video packet to be transmitted to 1320 bytes. A generalization of the video block fragmentation rule is

$$n \times 10\ kbyte = x \times 1320\ bytes + y \tag{1}$$

where n is the number of requested video blocks, x is the number of 1320-byte video packets, and y is the size of the smaller fragment in bytes (377, 497, etc.), as presented in Table 1.

5 Topological properties of SopCast

5.1 A two-layer SopCast overlay

The topology of the SopCast overlay network can be generated with peers as nodes and connections as links. The SopCast overlay is constructed with decentralized peers. We study the SopCast overlay as a two-layer architecture consisting of the *neighbor graph* G_N and the *video graph* G_V. Both graphs are formed with directed links. We define a peer as a node that is active in both downloading and uploading processes. The SP is not included in the two layers, because it only supports a number of children with video downloading. Notice that not all neighbors will share video streams with each other. A straightforward relation between the two layers is $G_V \subseteq G_N$.

The neighbor graph is formed by a set of nodes with neighbor relations. A link in the graph is denoted as a *neighbor link*. A neighbor link is established if a node pair is regularly sending keep-alive messages. If the keep-alive messages are not observed for some time, the link is considered to be inactive. The outgoing degree D_{out} of G_N defines the number of neighbors that a random peer has in the network. The video graph consists of peers that are transmitting video blocks. The incoming degree D_{in} of the video graph indicates the number of parents that a random peer has. The outgoing degree determines the number of children that an arbitrary peer supports. A *video link* is activated if there are video packets being transmitted from the parent to the child.

5.2 Reflecting the dynamic overlay

In a high churning network, such as SopCast, we are not interested in taking *instantaneous* snapshots. Because an instantaneous snapshot is taken at an exact time point (in the order of microseconds), thus capturing only a few nodes and

links. In this paper, we study the network dynamics by continuously taking network snapshots with the *duration* τ as time evolves and show them as a time series. A *snapshot* captures all participating peers and their connections within a particular time interval, from which a graph can be generated. The snapshot duration may have minor effects on analyzing slowly changing networks, such as Internet on Autonomous System (AS) level topologies. However, in a P2P streaming system, the characteristics of the network topology vary greatly with respect to the time scale of the snapshot duration, as addressed in [3].

We define an active (neighbor or video) link if two peers in SopCast are continuously transmitting keep-alive messages, or video packets. Often, we could notice temporary ceasing of the connection between two peers. After a period which ranges from several seconds to hundred seconds, they contact each other again. If the time period that two peers stop communication is in the order of hundred seconds, we can confirm a connection closure on the (neighbor or video) link. Whereas, due to transmission, or processing delay, or network congestion, a peer may send the keep-alive packets, or video packets with longer delay. Hence, such a link should be considered as active even though packet delivery is not observed for some small time.

To define the activation period of a neighbor link in SopCast, we have processed the traces and obtained the time interval between two consecutive keep-alive messages between all node pairs. From the parsed dataset, the corresponding probability density function (pdf) is plotted in Fig. 2 (left figure). This statistical analysis suggests that, with high probability (more than 95% of the cases), a peer sends two consecutive keep-alive packets to its neighbors within 2.5 seconds. Thus, we consider the threshold of $\tau_N \sim 2.5$ s as the activation period of a neighbor link. If two peers have not contacted each other within this interval, termination of a link connection is assumed. With the same approach, the pdf of the time interval between two consecutive video requests for all node pairs is plotted. Indicated in Fig. 2 (right figure), the activation period of a video link is around $\tau_V \sim 2.0$ s. If a child A requests video blocks from its parent B within 2.0 s from the previous request, the video link is considered to be active.

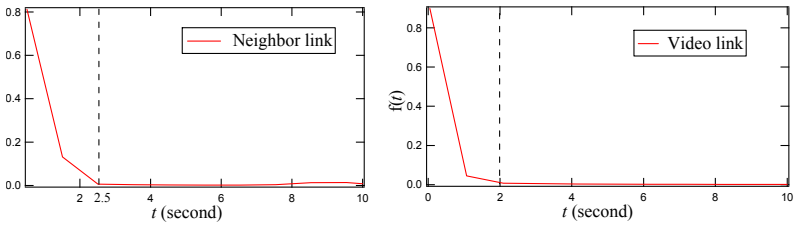

Fig. 2. Pdf of the activation period of the neighbor link (left) and the video link (right). Threshold of activation period for the neighbor link is $\tau_N \sim 2.5$ s, and$\tau_V \sim 2.0$ s for the video link.

5.3 Topology evolution of the neighbor graph

SopCast deploys certain principles to discover neighbors by exchanging peerlists. A peer is consequently expected to be able to contact, at least a portion of the neighboring peers in the network. We take snapshots of the neighbor graph during the time interval of $[0, T]$, with $T = 10$, 60, 300, 600 seconds respectively, so that all the participating peers and links that have once been active from the beginning of the experiment till the time T have been captured. All the historical neighboring connections are accumulated in these snapshots.

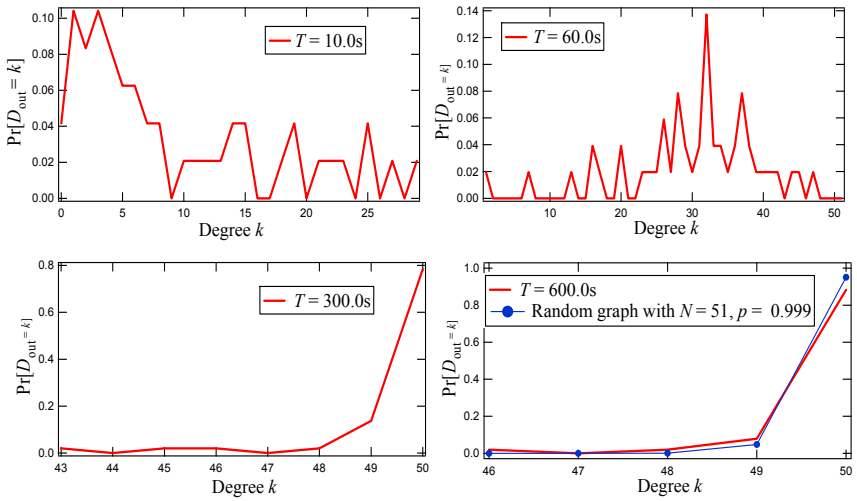

Fig. 3. Topology evolution of the neighbor graph as a function of time series. The outgoing degree defines the number of neighboring connections a node once maintained. In the bottom-right sub-figure, $Pr[D_{out} = k]$ is fitted with the degree distribution of a random graph with link density $p = 0.999$.

With the captured snapshots, four directed graphs are obtained. Multiple lines between two peers are removed. We plot the pdf of the outgoing degree (D_{out}) of the four graphs in Fig. 3. We notice that peers discover their neighbors quite fast in SopCast. After 300 seconds, the neighbor topology already converges to a full mesh, which means that SopCast peers have the ability to find almost all the other neighbors within a relatively short period of time. The distinct property of neighbor discovery is better illustrated in Fig. 4, in which the evolution of the average outgoing degree of all peers is plotted as a function of time. The average outgoing degree grows rapidly during the first 5 minutes of the experiment[11]. The standard deviation of the average outgoing degree is also shown. In the first

[11] Compared to the entire duration of the experiment (40 minutes), this period can be considered to be short.

several minutes, peers tend to behave differently. Some of them contacted with more neighbors, while some only meet a few. This is the reason that a large standard deviation appears at the beginning of the experiment. As time evolves, the activity of neighbor discovery gradually converges at different peers. After 5 minutes, the standard deviation has reduced substantially.

The activation period of a neighbor link was found to be 2.5 s in the previous subsection. By taking network snapshots with this interval, we could obtain some insights on the number of neighbors that a peer communicates during this period. The neighbor graph is examined during the time interval of [5 min, 35 min]. This is because, in the first 5 minutes, peers are trying to discover more neighbors, thus the peerlist may not be completely filled in. The last 5 minutes are also excluded, because two PlanetLab nodes came off-line unexpectedly. By taking snapshots every consecutive 2.5 s, 720 network topologies are obtained, which generate 28050 samples of the outgoing degree in total. The derived pdf of the outgoing degree is well fitted with a Gaussian distribution. The average number of contacted neighbors within 2.5 seconds is around 37.73. We also extended the snapshot duration to 10 seconds as a comparison. The average outgoing degree increases slightly to 41.60. This observation reveals relatively stable connections between peers. If a peer frequently removes its neighbors in the peerlist and substitutes them with new ones, the average outgoing degree of the neighbor graph would be larger than 41.60 in the longer snapshot interval of 10 s.

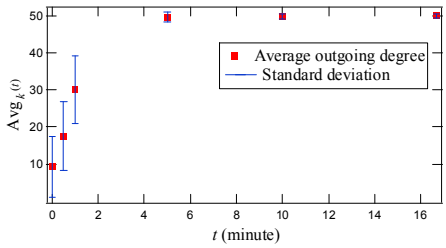

Fig. 4. The average outgoing degree of the neighbor graphs as a function of time.

5.4 Topology dynamics of the video graph

The SopCast protocol specifies a set of streaming delivery rules that are used to guide peers to actively connect to others, or allow other peers to establish connections to themselves. We aim to understand how SopCast coordinates peers to share video streams from the perspective of a network topology.

We study the video graph when peers have chances to contact with almost every neighbor in the network, e.g. after 5 minutes of the experiment in Fig. 3, because a peer is expected to keep the best performing neighbors in its peerlist and the network is considered to be more stable from this time. Hence, the

topology of the video graph is also examined during the time interval of [5 min, 35 min].

Recall that the activation period of a video link was found to be 2.0 seconds. We divide the time period of [5 min, 35 min] to sequential time slots of 2.0 seconds, resulting in 900 network snapshots of G_V. The 900 snapshots provide us with 32320 samples of the incoming degree (D_{in}), and 27697 samples of the outgoing degree (D_{out}). The incoming and outgoing degree distributions are obtained by making the histogram of the 32320 in-degree samples and 27697 out-degree samples respectively. In Fig. 5, SopCast peers only retrieve video packets from, on average, 1.86 neighbors during 2.0 seconds. In rare cases, a node may contact more than 10 peers to download streams. The observation suggests that there might be a threshold γ of the maximum number of parents that a node is allowed to connect with, most likely with $\gamma = 4$.

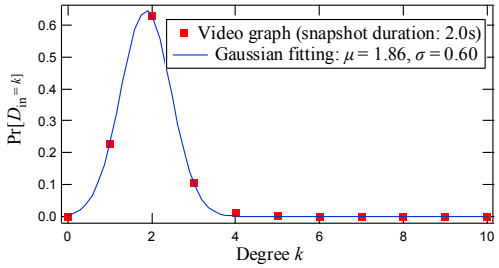

Fig. 5. Pdf of the incoming degree (number of parents) of the video graph on a linear scale. Snapshot duration of the video graph is set to 2.0 s. A Gaussian fitting describes the pdf of D_{in} well, which suggests us to model the incoming degree of G_V as a random graph with very low link density $p \ll 1.0$.

The outgoing degree distribution of the video graph shows different behaviors. A supernode structure emerges when looking at the curve plotted on a log-log scale in Fig. 6 (left). A *supernode* in a P2P network refers to a node that can offer good uploading bitrates and establishes many connections with different children. It seems that SopCast peers are allowed to select several parents to establish downloading connections, while they will only choose the ones that provide better uploading performance, which leads to the existence of supernodes. In this section, we only inspect the supernode structure in terms of node degree. In Section 5.5, we will show the supernode structure from the perspective of uploading throughput. Interestingly, $\Pr[D_{out} = k]$ on a log-lin scale also fits well with a line curve. A similar observation was found by Janic *et al.* [5]. We think the network size is not large enough to warrant a definite conclusion. However, with the PlanetLab network, a large networking scale, e.g. $N \gg 100$ was not possible during the period in which we performed the experiments. We

have carried out another experiment with 132 PlanetLab nodes, which gave us similar results as in Fig. 6.

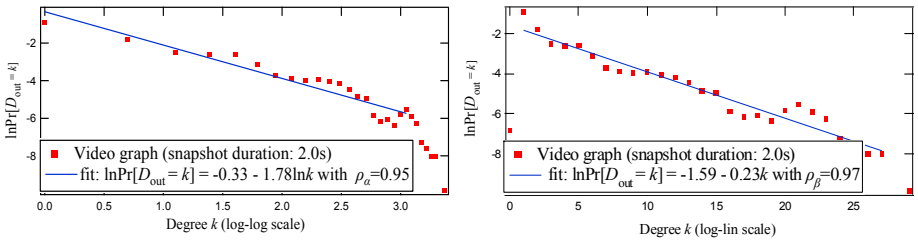

Fig. 6. The outgoing degree (number of children) distribution of the G_V on both log-log scale (left) and log-lin scale (right). The linear correlation coefficients ρ_α on the log-log scale and ρ_β on the log-lin scale are used to reflect the fitting quality.

In the following, we will substantiate our statement on the importance of the snapshot duration for reflecting the network properties. By increasing the snapshot duration to 10 s, a dominating power-law distribution of the outgoing degree is still observed. When enlarging the time scale of the snapshot duration to 100 s, the power-law behavior of D_{out} is replaced by a Gaussian distribution. The discrepancy of the distribution in D_{out} with snapshot duration of 100 s is mainly caused by the fact that a peer may support different children over time and these changes are accumulated in the captured snapshots. The divergency observed with different snapshot durations suggests us to model the video graph with respect to different time scales that are used to capture network snapshots. Besides, the accuracy of the obtained snapshots should be discussed carefully. In our case, we consider a snapshot duration of 2.0 seconds as the accurate parameter to characterize the essential properties of the video graph.

5.5 SopCast network load distribution

A critical issue of a peer-to-peer streaming system is to fairly distribute the network load to peers participating in the video delivery process. It is very likely that the so called tit-for-tat principle is not implemented in SopCast. A peer can request video packets from one of its neighbors, without offering any video blocks to this neighbor. For instance, we have noticed constant uploading from PlanetLab node *freedom.informatik.rwth-aachen.de* to *planetlab1.pop-mg.rnp.br*, but no uploading activity in the reverse direction.

In Fig. 7, we present the total downloading and uploading throughput at every single peer. Peers tend to have uniform downloading throughput, as the black dotted line shows. However, they behave differently regarding their uploading performance. There are several nodes downloading much less than others, which may be caused by bandwidth limitations[12] at these nodes.

[12] Bandwith of a PlanetLab node can be limited by PlanetLab administrator.

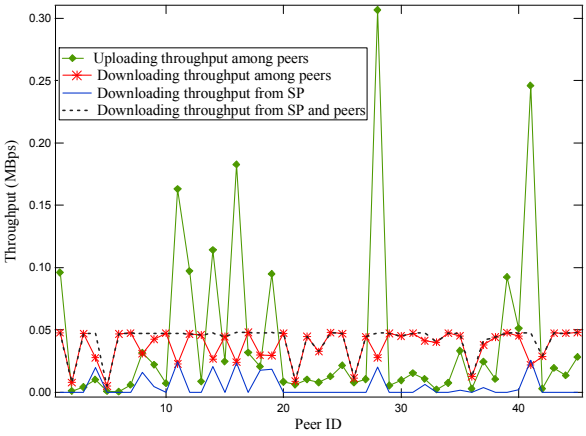

Fig. 7. Uploading and downloading throughput at every SopCast node during the entire experiment (40 minutes). The x-axis presents the identification number (ID) of each individual peer. The uploading throughput (downloading throughput) at a single peer is calculated based on the amount of data uploads (downloads) to all the children (parents) the peer has.

In SopCast, a node can download video streams either from the SP, or from other peers. We noticed that SopCast has limited the maximum number of children connected to the SP simultaneously to 11. The downloading throughput from decentralized peers and the SP at every single node are also presented in Fig. 7. We notice that the SP in SopCast does not change its children very often. During the entire experiment, only 15 nodes downloaded video blocks from the SP. These 15 peers act as the major force (supernodes) to provide video blocks to other peers and contribute the most in terms of uploading throughput. The burden of the SP is consequently alleviated.

We have indicated the existence of supernodes in terms of node degree and uploading throughput. We also found a strong correlation between the outgoing degree and the uploading throughput of a supernode, namely, a peer that establishes many connections with its children uploads more video blocks. Our observations suggest a hierarchical structure of the video graph. Although the SP performs a critical role in broadcasting the video streams, it is not the major resource from which peers can download video blocks. The peers that are directly connected to the SP act as the supernodes in SopCast. They support many children, and contribute to the largest uploading bandwidth in the network. The other peers download video blocks from the supernodes.

6 Conclusions

In this paper, we have performed a series of experiments in a controlled environment with a set of PlanetLab nodes running the SopCast application. Our

aim was to understand the mechanisms in SopCast and further to characterize the topological properties of this dynamic P2PTV overlay. Via passive measurements, we discovered the basic mechanism deployed in the SopCast protocol and modeled the SopCast network with a two-layer architecture.

Based on our measurements and analysis, our main conclusions are: 1) SopCast traffic can be categorized in control packets and video packets. The control packets fulfil different functionalities, which coordinate the operation of the SopCast application. The video packets deliver the TV content. 2) Communication between SopCast peers is decentralized. Peers discover their neighbors very fast. The constructed neighbor graph is considered to be resilient, since it is close to a full mesh. 3) Video delivery in SopCast is chunk-based. Each video chunk has equal length of 10 kbytes. A peer is free to request multiple blocks from its parent(s). 4) The incoming degree distribution of the video graph can be modeled as a random graph with very low link density $p \ll 1.0$. While a potential power law behavior is observed with the outgoing degree distribution of the video graph. A node with good uploading performance can act as a supernode in SopCast. A supernode sacrifices a large amount of upload bandwidth to its many children. 5) An adequate snapshot duration is important in understanding the actual topology changes of P2PTV networks. In SopCast, we have found $\tau_N \sim 2.5$ s, and $\tau_V \sim 2.0$ s as the reference snapshot duration in the neighbor graph and video graph respectively.

References

1. A. Sentinelli, G. Marfia, M. Gerla, L. Kleinrock, and S. Tewari, "Will IPTV Ride the Peer-to-Peer Stream", IEEE Communications Magazine, Vol. 45, Issue 6, pp. 86-92, June 2007.
2. D. Purandare and R. Guha, "An Alliance based Peering Scheme for Peer-to-Peer Media Streaming", Proc. of the Workshop on Peer-to-Peer Streaming and IP-TV, in conjunction with ACM SIGCOMM, Kyoto, Japan, Aug 27-31, 2007.
3. D. Stutzbach, R. Rejaie, and S. Sen. "Characterizing Unstructured Overlay Topologies in Modern P2P File-Sharing Systems", IEEE/ACM Transactions on Networking, Vol. 16, No. 2, April 2008.
4. A. Orebaugh, G. Ramirez, J. Burke, and L. Pesce, *Wireshark & ethereal network protocol analyzer toolkit (jay beale's open source security)*, Syngress Publishing, 2006.
5. M. Janic and P. Van Mieghem, "On properties of multicast routing trees", International Journal of Communication Systems, Vol. 19, No. 1, pp. 95-114, February 2006.
6. S. Ali, A. Mathur, and H. Zhang, "Measurement of Commercial Peer-to-Peer Live Video Streaming", Proc. of the Workshop In Recent Advances in Peer-to-Peer Streaming, August 2006.
7. T. Silverston and O. Fourmaux, "Measuring P2P IPTV Systems" , Proc. of Network & Operating Systems Support for Digital Audio & Video (NOSSDAV'07), June 2007.
8. X. Zhang, J. Liu, B. Li, and P. Yum, "DONet/Coolstreaming: A datadriven overlay network for live media streaming," Proc. of IEEE INFOCOM, March 2005.