LOW LATENCY VIDEO STREAMING OVER PEER-TO-PEER NETWORKS

Eric Setton, Jeonghun Noh and Bernd Girod

Information Systems Laboratory, Department of Electrical Engineering
Stanford University, Stanford, CA 94305-9510, USA
{esetton, jhnoh, bgirod}@stanford.edu

ABSTRACT

We study peer-to-peer multicast streaming, where a source distributes real-time video to a large population of hosts by making use of their forwarding capacity rather than relying on dedicated media servers. We present a distributed streaming protocol which builds and maintains multiple multicast trees. The protocol is combined with an adaptive scheduling algorithm which ensures packets destined to a large number of peers, or particularly important to decode the video, are sent in priority. Experiments carried out over a simulated network of up to 3000 peers illustrate the performance of the protocol. For low latency video streaming, the prioritization algorithm offers performance gains, especially for large audiences and low latencies.

1. INTRODUCTION

In live peer-to-peer (P2P) streaming, a video stream is transmitted to a large population of viewers, through the use of the uplink bandwidth of participating peers. Unlike server-based streaming systems, where the number of media servers required to serve an audience grows linearly with the number of viewers, this approach is self-scaling as the number of peer "servers" and peer "clients" increases at the same rate. Therefore, P2P multicast is a compelling way to lower the cost of large scale streaming or to address the problem of flash crowds, which can overwhelm a content delivery network.

To achieve the same success as P2P file transfer networks, which represent, today, over 50 % of total Internet traffic [1], P2P streaming systems should achieve high and constant video quality, as well as low startup latencies, and require no dedicated infrastructure. Three factors make this a difficult task. First, the access bandwidth of the peers is often insufficient to support high quality video. Second, the peers may choose to disconnect at any time and make for a highly unreliable and dynamic network fabric. Third, unlike in client-server systems, packets often need to be relayed along long multi-hop paths, each hop introducing additional delay, especially when links are congested. This unique set of challenges explain why early implementations such as ESM [2] and PPLive [3] fall short of the goals. Although these implementations constitute very exciting progress and demonstrate the feasibility of large scale P2P streaming, they all suffer from long latencies, usually on the order of minutes, and unstable video quality.

One way of improving the performance of P2P video streaming systems is to break away from the common practice which focuses on designing better protocols, while ignoring the properties of the transmitted data stream. As an alternative, adaptive algorithms where the encoding and the streaming is adapted to the network protocol can and should be considered. Cross-layer-designed P2P streaming architectures were presented in [4] where the authors combine path diversity and multiple description coding, and in our prior work [5, 6], where video coding and packet scheduling is adapted to a multicast tree-building protocol. The purpose of this article is to analyze further the performance of our P2P protocol and of a streaming algorithm which combines information about the video content and information about the underlying multicast trees to prioritize packet transmissions efficiently.

In the next section we describe our distributed P2P protocol, which builds and maintains multiple multicast trees. Results collected over a simulated network of up to 3000 peers reflect the scalability of the protocol and the benefits of employing multiple multicast trees. In Section 3, we explain how to prioritize transmissions by favoring in particular packets destined to a large number of peers, or particularly important to decode the video. Finally, we analyze experimental results which illustrate the benefits of the approach for low latency streaming.

2. DISTRIBUTED P2P PROTOCOL

The control protocol enables a source to distribute a video stream to a population of peers via application-layer multicast. It is completely distributed, except for an approximate list of connected peers stored and updated by the source.

This work was supported, in part, by a gift from Hewlett-Packard Laboratories, Palo Alto, CA.

The video source and the peers are connected via multiple multicast trees which are constructed dynamically. The source is the root of all trees and the trees are built and maintained mostly independently. The video stream is transmitted along the different multicast trees in round-robin fashion. Hence, peers need to join each of the multicast trees to decode and play out the video successfully. If a peer leaves the session, the multicast trees are dynamically reorganized as affected peers attempt to reconnect by establishing links to other parents.

2.1. Joining

When a peer wishes to join the multicast, it contacts the source and obtains a partial list of peers participating in the streaming session. Connections to the different multicast trees are established by determining which of these peers has enough bandwidth to support an additional child. When there are several parent candidates, the one closest to the source is chosen to minimize the height of the trees being built. Ties are broken by choosing, when possible, different parents for different multicast trees. This increases the diversity and improves the retransmission algorithm outlined in the following. In this paper, we ignore any Network Address Translator or firewall issue which may limit connectivity. Although these problems need to be addressed for a real Internet implementation (see e.g. [2]), they are not directly relevant to the scope of this paper and efficient solutions have been proposed [7].

2.2. Node disconnection

When a host leaves, it stops forwarding video packets and is unresponsive to probing. Peers detect disconnections by monitoring the state of their parents, and attempt to rejoin when they detect traffic interruptions. A peer will first try to rejoin the tree it has been disconnected from through one of its other parents. If this fast recovery mechanism fails, the peer will then contact the source to get a new list of connected peers, and follow the join procedure.

While the peer reconnects, retransmission requests are issued over the other multicast trees to recover missing video packets. This approach is different from forward error correction (FEC) which introduces redundancy in the stream and does not require the use of a feedback channel. Unlike FEC, adaptive retransmission requests can be implemented without any assumption on the error rate of the channel. Moreover, temporally correlated loss patterns which characterize peer disconnections would cause FEC to fail or require large overhead and a waste of network bandwidth. In [5] and [6], we explain how retransmission requests can efficiently be managed by a low complexity algorithm which reduces the additional strain on the other multicast trees, by requesting in priority the most important packets in terms of video quality.

2.3. Simulation setup

We evaluate the performance of the protocol over a network simulated in ns-2 [8]. The backbone links are sufficiently provisioned so that congestion only occurs on the access links. The latency of each link is 5 ms, and the diameter of the network is 10 hops. Losses are only due to congestion and queue overflow, and transmission errors due to the presence of ISP boundaries or potential wireless last-hop links are ignored. The number of peers participating in the multicast varies between 10 and 3000.

The bandwidth of the peers reflects today's popular network access technology. The bandwidth distribution is given in Tab. 1. and is similar to the findings reported in [9]. In the experiments, peers join and disconnect from the multicast session randomly. The average time for which peers tune in to the multicast is around 5 minutes.

Downlink	Uplink	Percentage
512 kbps	256 kbps	56%
3 Mbps	384 kbps	21%
1.5 Mbps	896 kbps	9%
20 Mbps	2 Mbps	3%
20 Mbps	5 Mbps	11%

Table 1. Distribution of peer bandwidth.

The video stream produced by the source is encoded with H.264 [10] at a constant quality; the encoding rate is approximately 250 kbps. Each video frame is packetized into UDP packets. Frames exceeding the maximum transmission unit size are fragmented before packetization.

2.4. Performance of the protocol

Figure 1 illustrates the performance of the protocol in terms of scalability. As expected, the total amount of control and video traffic, given on the left vertical axis of the figure, increases linearly with the number of peers. However, the percentage of traffic due to control, indicated with a solid line and measured on the right vertical axis of the figure, remains constant, and only represents 2% of the total traffic. This demonstrates the scalability of the protocol within the limits of this experimental setup.

The results shown in Fig. 2 illustrate the benefits of using multiple multicast trees. As the number of trees increases, the rate of the video sub-streams forwarded along each tree decreases linearly. Hence, the amount of free uplink bandwidth required to support an additional child on any particular tree is smaller; this finer granularity leads

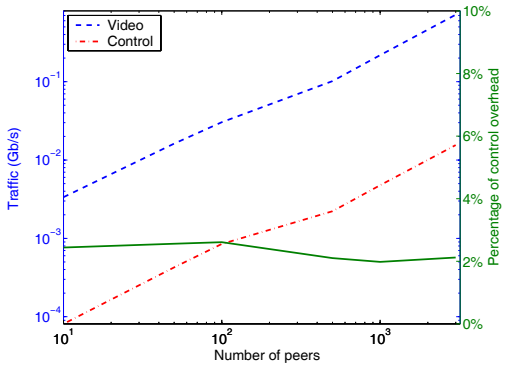

Fig. 1. Aggregate traffic and control overhead for different numbers of peers and 4 multicast trees.

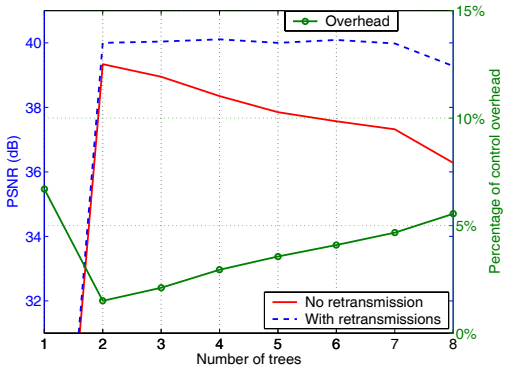

Fig. 2. Average video quality and control overhead for 300 peers and varying numbers of multicast trees.

to better use of the available network bandwidth. This explains why the network cannot achieve good performance with only 1 multicast tree. In this case, more than half of the peers are free-riders, as their throughput does not allow them to forward the video stream. For 2 or more trees, finer granularity leads to additional available uplink resources. In particular, there are no more free riders and the available resources are sufficient to sustain high video quality. This is the prevalent factor in determining a suitable number of trees. The results in Fig. 2 show that when more multicast trees are used, the control traffic needed to build and maintain the trees increases, as shown by the dotted line and measured on the right vertical axis. In addition, the increased frequency of parent disconnections causes more losses which results in lower video quality when no retransmissions are allowed. A high video quality can, however, be maintained by using retransmissions.

3. PRIORITIZED TRANSMISSION

Relaying traffic over the uplink of the peers may lead to congestion on the multi-hop path separating the source from any particular peer. In particular, because the rate of a video stream often varies, a peer may sometimes lack the resources to forward all the data expected by its descendants. Scheduling can help maintain video quality in the instances when a peer has to drop some packets to ensure timely delivery of the more significant portion of the streams. To our knowledge, although scheduling algorithms for media streaming have received an increasing amount of attention (see in particular work based on the seminal article by Chou and Miao [11]), little or no work has studied this topic in the context of P2P.

3.1. Congestion-distortion optimized prioritization

In related work [6], we presented a scheduling algorithm named CoDiO which performs congestion-distortion optimized packet scheduling. CoDiO not only determines in which order to send packets destined to a particular peer, but also, how to prioritize among the different descendants of a peer. This prioritization algorithm bases its decisions on the "importance" of each enqueued packet. This metric reflects the distortion reduction associated with decoding a particular packet, and must be adjusted according to the number of descendants in the multicast tree that would be affected by the loss or late arrival of this packet. Hence, the scheduler adapts its decisions to the video content and to the structure of the underlying multicast trees.

CoDiO determines which is the next most important packet by comparing the importance of each enqueued packet. For a packet number n, addressed to peer m the importance is expressed:

$$I(n,m) = D(n) * (NumDescendants(m) + 1) \quad (1)$$

Fig. 3. GOP encoding structure.

In (1), $NumDescendants(m)$ represents the number of peers to which packet n will be forwarded after reaching peer m, this number is collected by the control protocol when information is exchanged between neighboring peers to maintain the multicast trees. The quantity $D(n)$ reflects the sensitivity of the video quality to the reception of the packet. As the peer does not collect detailed rate-distortion information about the video stream it is transmitting and as the exact state of the reception buffer of its descendants

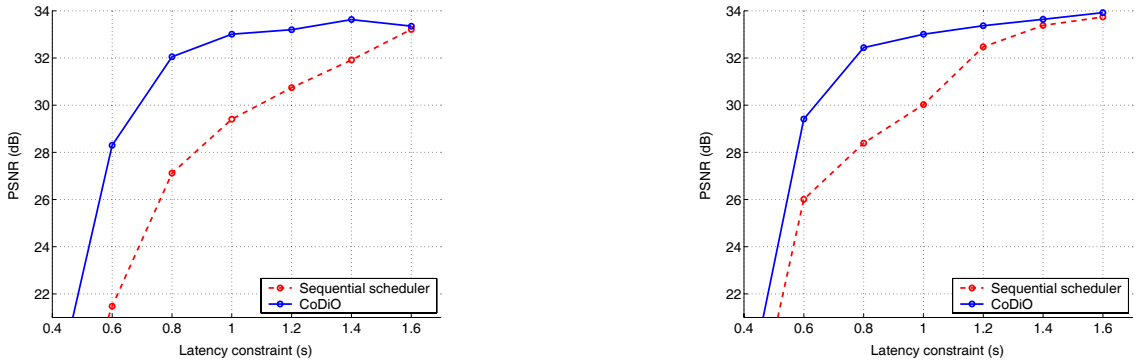

Fig. 4. Average video quality for 300 peers (left) and 75 peers (right), as a function of the latency constraint for *Foreman*

is not known either, this sensitivity needs to be approximated. We choose to express $D(n)$ as the number of frames which will be affected if the frame packet n belongs to is not decoded correctly. Therefore, for the encoding structure shown in Fig.3, the importance of the different frames is 19, 15, 11 and 7 for the I frame and for the subsequent P frames and 1 for each B frame.

To mitigate congestion, transmissions are spaced based on the time needed for the last packet to traverse the uplink, while reserving a fraction of the link for control traffic.

3.2. Experimental results

To analyze the benefits of adaptive scheduling we compare CoDiO to a sequential scheduler, in the simulation setup described in Sec. 2. The sequential scheduler relays the packets it has received, as long as they are not past due, without changing their order. Results are collected for different latency constraints[1]. When the latency constraint is relaxed, both schedulers have similar performance, as illustrated in Fig. 4[2]. In this case, reducing the end-to-end delay for the more important packets has little influence. When the latency constraint becomes tighter, CoDiO maintains high video quality while the performance of the sequential scheduler degrades. In this experiment, the maximum tolerable latency is reduced by up to 50%. The range of latencies over which scheduling offers a sizeable performance improvement depends on the number of peers participating in the session. For larger audiences, it is essential to prioritize packets destined to large population of peers.

4. CONCLUSION

In this paper we analyze the performance of a multicast streaming protocol which builds and maintains muticast trees to transmit live video to a large population of peers. Experiments carried out over a simulated network of up to 3000 peers show the scalability of the protocol and illustrate the benefits of using multiple multicast trees. We also describe an adaptive scheduling algorithm which improves the performance of low latency P2P video streaming, by favoring, in particular, critical video frames and peers with a large set of descendants.

[1]The latency constraint denotes the time between the instant a packet is generated by the source and its playout deadline

[2]Video quality is measured as the Peak Signal to Noise Ratio (PSNR) and expressed in decibel

5. REFERENCES

[1] "Global traffic levels," *http://www.cachelogic.com/research/2005_slide12.php, seen on Apr. 18 2006*.

[2] Y. Chu, A. Ganjam, T. Ng., S. Rao, K. Sripanidkulchai, J. Zhan, and H. Zhang, "Early experience with an internet broadcast system based on overlay multicast," *USENIX Annual Technical Conference, Boston, MA*, pp. 1283–1292, June 2004.

[3] "PPLive," *http://www.PPLive.com, seen on Aug. 28 2005*.

[4] V. Padmanabhan, H. Wang, P. Chou, and K. Sripanidkulchai, "Distributing streaming media content using cooperative networking," *Proceedings NOSSDAV'02, Miami, USA*, May 2002.

[5] E. Setton, J. Noh, and B. Girod, "Rate-Distortion optimized video peer-to-peer multicast streaming," *Workshop on Advances in Peer-to-Peer Multimedia Streaming at ACM Multimedia*, pp. 39–48, nov 2005.

[6] E. Setton, J. Noh, and B. Girod, "Congestion-distortion optimized peer-to-peer video streaming," *Internet conference on image processing (ICIP), to appear*, 2006.

[7] B. Ford, P. Srisuresh, and D. Kegel, "Peer-to-peer communication across network address translators," *Proceeding USENIX, Anaheim, USA*, Apr. 2005.

[8] "The Network Simulator - ns-2," *www.isi.edu/nsnam/ns/, seen on Aug. 28 2005*.

[9] K. Sripanidkulchai, A. Ganjam, B. Maggs, and H. Zhang, "The feasibility of supporting largescale live streaming applications with dynamic application endpoints," *Proceedings SIGCOMM'04, Portland, USA*, Aug. 2004.

[10] ITU-T and ISO/IEC JTC 1, *Advanced Video Coding for Generic Audiovisual services, ITU-T Recommendation H.264 - ISO/IEC 14496-10(AVC)*, 2003.

[11] P. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," *Microsoft Research Technical Report MSR-TR-2001-35*, Feb. 2001.