INVITED
PAPER

Peer-to-Peer Live Multicast: A Video Perspective

Systems designed to provide television quality viewing in peer-to-peer computer networks, and the tools to measure their effectiveness, are now being developed.

By Eric Setton, *Member IEEE*, Pierpaolo Baccichet, *Member IEEE*, and Bernd Girod, *Fellow IEEE*

ABSTRACT | Peer-to-peer multicast is promising for large-scale streaming video distribution over the Internet. Viewers contribute their resources to a peer-to-peer overlay network to act as relays for the media streams, and no dedicated infrastructure is required. As packets are transmitted over long, unreliable multipeer transmission paths, it is particularly challenging to achieve consistently high video quality and low end-to-end delay. In this paper, we focus on error-resilient transport for peer-to-peer video streaming. The algorithms we describe are representative of three broad categories of robust video streaming schemes: forward error correction, multiple descriptions, and prioritized automatic repeat request. We analyze how these techniques can be employed for live peer-to-peer multicast and discuss their relative merits. Our results show that significant gains can be obtained when systems are designed to adapt to the encoding structure of the video streams they are transmitting. They also reveal the importance of avoiding congestion at every peer participating in the multicast to obtain a low-latency system. Finally, we provide insights as to which are the important metrics to compare different peer-to-peer streaming systems.

KEYWORDS | Error resilience; multicast; peer-to-peer; video streaming

I. INTRODUCTION

In live peer-to-peer (P2P) video multicast, a stream is transmitted to a large population of clients, utilizing the uplink bandwidth of participating peers. Similar to popular file transfer networks, such as BitTorrent (or any client running the BitTorrent protocol, e.g., Azureus), Gnutella, eDonkey, or Kazaa, media delivery is accomplished via a distributed protocol that lets peers self-organize into distribution trees or meshes. The striking difference is that data transfer happens in real time to provide all connected users with a synchronous, TV-like viewing experience. Compared to content delivery networks, this type of distributed system is appealing, as it does not require any dedicated infrastructure and is self-scaling as the resources of the network increase with the number of users. The first generation of P2P multicast systems, which established the feasibility of the approach, has focused on enabling the largest possible set of peers to connect to application-layer multicast sessions.

To achieve the same success as P2P file transfer networks, which represent, today, more than 60% of total Internet traffic, P2P live video streaming systems must achieve reliable 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. Secondly, the peers may choose to disconnect at any time giving rise to a highly unreliable and dynamic network fabric. Thirdly, unlike in client–server systems, packets often need to be relayed between several peers, with each hop introducing additional delay, especially when links are congested.

In this paper, we focus on the unique issues of video transport over P2P networks. Thus, we build upon, and complement, the paper by Liu *et al.*, [1] appearing in this Special Issue. We show that, to achieve better performance in P2P video streaming systems, application-layer multicast and video transport should be considered jointly. Streaming algorithms must be tailored to the content and to the distribution topology to provide the best experience to the largest number of users. While network-adaptive video encoding and streaming has been studied extensively for server–client systems, this approach is still novel for P2P streaming systems. We expect it to be of central importance for the second generation of P2P video

Manuscript received October 15, 2006; revised September 24, 2007.
E. Setton is with Dyyno Inc., Palo Alto, CA 94303 USA (e-mail: eric@dyyno.com).
P. Baccichet is with the Max Planck Center for Visual Computing and Communication, Stanford University, Stanford, CA 94305 USA (e-mail: bacci@stanford.edu).
B. Girod is with the Information Systems Laboratory, Department of Electrical Engineering, Stanford University, Stanford, CA 94305 USA (e-mail: bgirod@stanford.edu).
Digital Object Identifier: 10.1109/JPROC.2007.909925

multicast. Our purpose is to show how content-adaptive distributed scheduling, in the context of P2P distribution over multiple complementary multicast trees, may enhance video quality, increase robustness to errors, and reduce end-to-end latency.

In Section II, we present an overview on advances in error-resilient video streaming related to P2P multicast. In Section III, we discuss the error resilience of video streams. In particular, we explain how different portions of a stream have a different impact on the decoded video quality. This is usually overlooked by today's P2P streaming systems, which often consider media streams as generic data sources. We describe how several transport techniques for error-resilient P2P video streaming can be implemented in Section IV. The first one, forward error correction (FEC), is a classic channel coding technique and can be elegantly combined with P2P streaming over multiple trees. The second, multiple descriptions (MDs), is also often combined with multipath transmission and is a natural consideration for tree-based P2P video streaming. Lastly, we present a scheduler, based on feedback and retransmissions, which uses congestion-distortion optimization (CoDiO) to prioritize packet transmission. This scheduler, which has been the subject of our recent work, takes into account the varying importance of packets in terms of impact on the video as well as the position of different peers in the multicast trees to decide which packets to transmit or retransmit in priority. In Section V, we compare the performance of these different schemes. Our experimental results are collected over a simulated network with thousands of dynamic peers all running the Stanford peer-to-peer multicast (SPPM) protocol, specifically targeted at low-latency streaming. Finally, in Section VI, we discuss the insights gained from recent deployments of SPPM and propose a set of metrics that can help compare different P2P multicast systems.

II. BACKGROUND

Despite the growth in availability of broadband technology, and progress in video compression, the quality of video streaming systems on the Internet is still not on par with standard-definition television (SDTV) and even further from high-definition television (HDTV). This is due to the best effort nature of the network, which does not offer any guaranteed quality of service as the throughput, delay, and losses may vary unexpectedly. The advent of wireless access adds challenges owing to interference, shadowing, and mobility. It is a daunting task to achieve consistently high video quality, low startup latencies, and short end-to-end delays in such environments. Recent advances in the field of video streaming focusing, notably, on adaptive video coding and error resilience are described, in particular, in [2]–[5]. In the following, we focus on advances related to P2P video streaming.

Most of the work on P2P live streaming systems, initially made popular by Chu *et al.* [6], focuses on protocol design [7] and does not consider advanced error-resilient transport schemes. Many proposed systems rely on distributed protocols to construct one or several multicast trees to distribute the stream between the media source and the different users [8]–[11]. Another approach lets peers self-organize in a mesh and request different portions of the video from their neighbors, with no particular emphasis on the structure of the distribution path [12]–[14]. Along with such early research experiments, many applications have appeared on the Internet, such as PPLive, PPStream, TVU networks, and Zattoo. All these implementations constitute very exciting progress and demonstrate the feasibility of large-scale P2P streaming. However, they typically suffer from long startup delays (possibly on the order of minutes). This is a consequence of using protocols, schedulers, and encoding structures that have not been targeted at low latency. In addition, their unstable video quality does not make for a high-quality viewing experience and is a result of insufficient robustness to transmission errors or peer disconnections.

To help combat losses due to unreliable peers, a media stream may be protected with channel coding. One of the most popular ways of achieving this is to apply FEC across the different packets of a compressed media stream, notably with Reed–Solomon codes. In this way, a receiver can recover the encoded stream from any large enough subset of packets. For video streaming, the priority encoding transmission (PET) scheme proposed in [15] is a popular way to provide unequal error protection (UEP) of different layers of a scalable video representation [16]–[18]. Optimizing the bit rate and the amount of protection of the different layers is studied in [19]–[22]. FEC can also be combined with data partitioning, which separates the stream into different segments and prioritizes important information such as headers and motion vectors [23], or it can be used to protect a region of interest using in particular the new error resiliency tools provided by H.264/AVC [24], [25]. Several papers have noted that FEC could be advantageously combined with P2P video streaming, e.g., [26] and [27]. In Section IV, we describe an approach that uses FEC for error resilience and discuss its merits.

Path diversity can also help improve the overall performance of a P2P streaming system. A sender may, for example, select the best end-to-end network path in terms of a variety of metrics, which include bandwidth, losses, or delay, or distribute a media stream along different routes. Although today's routers do not support source routing between two end hosts, path diversity can easily be obtained across the overlay topology formed by the members of a P2P multicast session. As losses are often temporally correlated along each path, splitting video packets between different independent routes is a way to protect the bitstream from consecutive losses, which can have a dramatic impact on decoded video quality [28]. This technique is often combined with multiple description coding (MDC) to send independently decodable streams over different paths [29]–[34]. When the probability of simultaneous losses on the paths is low, the error resilience increases at the cost of lower compression efficiency. For video coding, multiple descriptions can be obtained, for example, by temporal or spatial sampling, e.g., [35] and [36], or by using different transforms and quantizers [37]–[41]. Many papers advocate the use of MDC in P2P video streaming, e.g., [10], [26], [27], and [42]–[44], where different descriptions are transmitted over different application-layer multicast trees. In this context, the number of trees is an important design parameter since compression efficiency decreases with the number of descriptions, a concept often overlooked in the P2P video streaming community. In this paper, the multiple description encoding scheme we analyze in Section IV is obtained by applying UEP to the different portions of a compressed video stream. This has the advantage of not requiring a particular video encoder, since this operation can occur after the compressed video stream has been generated.

When the statistics of the path(s) are unknown, techniques based on automatic repeat request (ARQ) [45], which adapt to feedback, are widely employed for error recovery. This leads to the general question of finding the best way to schedule transmissions and retransmissions of the delay-constrained packets of an encoded stream. This problem is addressed, in a client–server context, through a Markov chain analysis in [46], but the exponentially growing search space limits the practicality of the scheme, for which heuristics have been suggested [47]. In [48], Chou and Miao suggest a framework, which has received significant attention in the video streaming community, for solving this problem through rate-distortion optimization (RaDiO). The aim of this approach is to find an optimal schedule for the packets of a stream that minimizes the Lagrangian cost $D + \lambda R$, where D represents the expected distortion and R is the transmitted rate. The RaDiO framework has been extended by Kalman *et al.* to include the impact of error concealment that better reflects the properties of video streams [49]. One limitation of RaDiO is its computational complexity. As noted in [50], this optimization problem is NP-hard and can be cast as a variation of the classic knapsack problem. Another limitation is that the work in this area has considered overprovisioned networks and ignored the impact of the transmitted stream itself on end-to-end delay. In our work [51]–[54], we have shown how some of these limitations may be overcome through the use of CoDiO scheduling. Different from rate-distortion optimization, CoDiO scheduling determines which packets to send, and when, to maximize decoded video quality while limiting network congestion.

This is particularly important in the context of P2P networks where there is no dedicated infrastructure and the receivers act as potential relays [55]. In this scenario, each of the participating peers has limited uplink bandwidth and can potentially serve a large number of subsequent receivers. Hence, it is particularly important to limit the congestion created on the uplink of the peers in order to keep low end-to-end delay between the multicast source and each of the peers. In addition, scheduling algorithms should be sufficiently simple to let peers process and prioritize the flow of incoming media packets in real-time, even as their number of descendants grows. In Section IV, we describe in more detail how CoDiO scheduling can be performed in the context of P2P streaming networks, as was first proposed in [56] and [57]. We analyze the performance of this cross-layer designed scheme, which incorporates information from the application layer such as the impact of different packets on video distortion, and information from the P2P topology to favor the most important peers.

III. ERROR RESILIENCE OF VIDEO STREAMS

In this section, we review the principles of video compression and describe how the encoding structure employed by today's widespread codecs leads to varying levels of importance in the frames that compose a compressed video stream.

Video signals are usually compressed to reduce the bit rate required for storage and transmission. In order to achieve high compression gains, encoders exploit statistical redundancy that exists in natural video signals. We show the basic components of a hybrid video encoder in Fig. 1. The input video signal is predicted from previously transmitted information available both at the encoder and the decoder, and the prediction error is compressed, typically with a transform coder operating on a block-by-block basis. The prediction can be based on information in

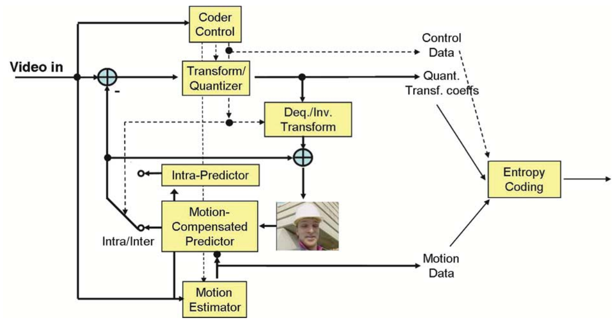

Fig. 1. ***Diagram of a motion-compensated hybrid video coder according to H.261, MPEG-1, MPEG-2, H.263, MPEG-4, or H.264/AVC standards. The intra–inter switch controls whether spatial or temporal prediction is used for compression. Dependency between frames is introduced via the motion-compensated inter frame prediction when P frames and B frames are encoded.***

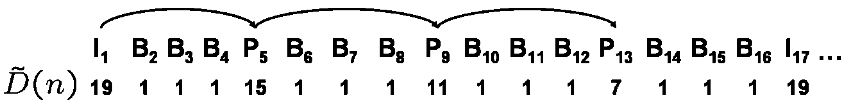

Fig. 2. ***Encoding structure of a group of pictures. Video frames are denoted by their type I, P, or B and are numbered according to their display order. The video importance of the frames $\tilde{D}$ is also represented.***

other frames ("motion-compensated predictor") or in the same frame ("intra predictor"). Depending on the type of prediction allowed, we distinguish three types of coded frames: intra (I) frames do not use temporal prediction but only intra prediction; predicted (P) frames use only one previously encoded frame as a reference; and bidirectionally predicted (B) frames combine prediction from two reference frames.[1] In general, I frames produce a much larger bit rate than P frames. The best coding efficiency can be achieved by using B frames. The residual signal after prediction is transformed in the frequency domain and quantized. Finally, entropy coding techniques, like context-based variable length coding or arithmetic coding, are applied to compress the syntax elements representing the video signal, which include motion vectors, coding modes, and quantized transform coefficients.

Higher compression efficiency makes the signal more susceptible to transmission errors. Even the corruption of a single bit in the compressed stream may preclude the decoding of a video syntax element. Since context based entropy coding is used, such an error will affect all the following syntax elements until a resynchronization marker is encountered. In addition, error propagation may occur within a frame, when a corrupted pixel value is used for prediction of adjacent pixels. Finally, regions of an image that cannot be correctly decoded create artifacts that are propagated over several consecutive frames, due to temporal prediction. Error propagation will continue until the next I frame is successfully decoded, since this type of picture does not depend on previously encoded pictures.

The different types of frames that compose the video suggest that not all video packets share the same perceptual importance. A coded group of pictures (GOP) with the chain of temporal dependencies between the frames is shown in Fig. 2 as an example. For this encoding structure, B frames depend on their neighboring I and P frames. Please note that the corresponding prediction arrows are not represented in the figure for clarity. For frame n the importance $\tilde{D}(n)$, also represented in Fig. 2, reflects the total number of frames that would be affected if the frame is not decoded correctly. The importance of an I frame is 19, as its loss would affect the 16-frame GOP as well as the three preceding B frames. The importance of the different P frames is 15, 11, and 7, depending on their place in the GOP, and the importance of each B frame is one. The importance function clearly illustrates that different pictures, depending on their type and their position in the group of pictures, have a widely varying effect on the decoded video quality of the stream.

To illustrate this effect further, we show in Fig. 3 the decoded video quality, measured as the mean square error (MSE) distortion, resulting from decoding a sequence of 296 frames, encoded at 30 frames per second, where a varying number of frames has been dropped. Results are shown for two different sequences compressed at around 300 kb/s, and although some variations are visible from sequence to sequence, the curves follow the same general trend. The average distortion value varies from sequence to sequence and reflects the different levels of activity of the clips. Three frame-dropping schemes are compared: the first one drops frames in increasing level of video importance, the second, in the reverse order; and the third randomly. Video importance is given by the function $\tilde{D}$ depicted in Fig. 2. The results for the three frame-dropping schemes show that for the same number of missing frames, the decoded video quality varies widely. Dropping an I frame in this example results in decoding errors in a total of 19 frames and is perceived by the user as freezing a decoded picture for over half a second. In addition, in the absence of their corresponding I frame,

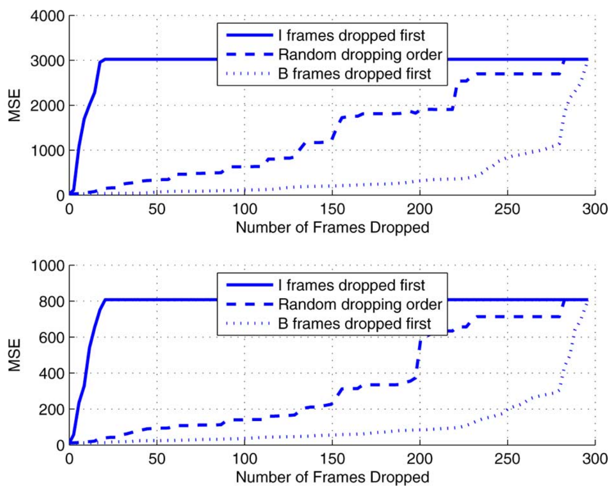

Fig. 3. ***Mean square error resulting from dropping different number of frames in different orders, for different sequences encoded at 300 kb/s. (Top) Foreman and (bottom) News.***

[1]Please note that these restrictions are required, e.g., in MPEG-1 and MPEG-2. The most recent H.264/AVC standard [58] is much more general and allows but does not mandate I, P, and B frames as described here.

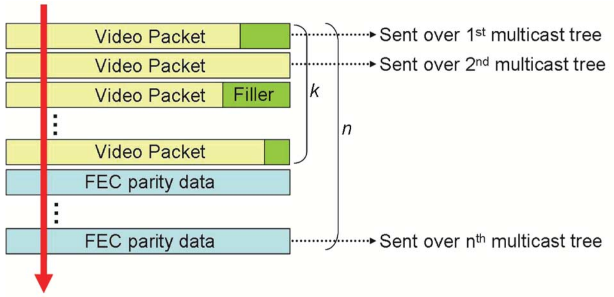

Fig. 4. ***An (n, k) Reed–Solomon code applied to k video packets for transmission over n multicast trees.***

other frames in a GOP are useless. This explains why the MSE reaches its maximum value for such a low number of frames, in Fig. 3, after all the I frames are dropped. This differs from the results of the other frame-dropping scheme, where frames are dropped following their order of importance. In this case, dropping B frames causes decoding errors where only one picture at a time is frozen, which results in a moderate and progressively increasing decoded distortion. Dropping the P frames from the end of the GOP to the beginning increases the MSE as a consequence of increasing the length of the intervals during which freezes occur. Finally the distortion reaches its maximum value when I frames are dropped. Not surprisingly, the random dropping scheme outperforms the scheme where I frames are dropped first and is less efficient than the scheme where frames are dropped following their order of importance.

This analysis clearly illustrates that a video transmission system, and in particular a P2P multicast network, should serve, with higher priority, packets belonging to I and P frames and with lower priority B frames. In addition, when application-layer information can be collected on the state of the decoder, it is a waste of the network resources to transmit packets that are undecodable due to the absence of their reference frames. This affects in particular the order in which retransmissions should occur. Finally, it is interesting to note that the length of the GOP is also an important design parameter for a P2P video streaming system. Indeed, as no frame can be displayed before an I frame is successfully decoded, the length of the GOP determines the minimum achievable latency, i.e., the minimum time a user will have to wait before a picture can be displayed. For P2P TV systems targeting latencies on the order of 1 s, the maximum compression efficiency will be reduced due to the need to insert I frames at short periodic intervals.

IV. ROBUST P2P STREAMING

In this section, we present three different types of error-resilient transport schemes for P2P multicast. We consider a P2P video streaming system that relies on multiple multicast trees rooted at the source to distribute a media stream in real time to a large population of peers. Each peer runs a distributed control protocol to establish and maintain connections to the different trees. Although the experimental results we present in Section V are for the SPPM protocol, the techniques we present, in this section, apply to general tree-based P2P multicast systems.

We consider a live transmission scenario where packets become available at the source progressively. To obtain a viewing experience comparable to broadcast television, the different peers play out the media stream synchronously. Hence, a packet has a limited amount of time, often called the *playout deadline*, between the moment it is made available to the multicast source and the moment it should be decoded by the peers. Beyond this time, for all the scenarios we consider, packets are discarded by the peers. In particular, late packets are not forwarded down the multicast trees, and resulting losses are concealed by the video decoder.

The multicast source is responsible for packetizing the media stream, prior to transmission. As video is transmitted to the different peers over multiple paths, out-of-order arrivals are expected. Packet headers are therefore necessary to provide enough information to reconstitute the media stream before decoding. Video packet headers such as provided by real-time transport protocol [59] are necessary to the decoder to identify packet numbers, video frame numbers, total number of packets in a frame, and the place of a particular packet among them, as well as the playout deadline of the different packets.

Aside from random transmission errors that happen occasionally, error patterns observed on P2P video streaming systems based on multicast trees are often bursty. This reflects the interruption that occurs when a peer disconnects unexpectedly from a multicast session and stops forwarding the media stream to its descendants. As well-designed systems tend to form arrangements where peers receive data from different neighbors on each multicast tree, these types of events tend to create losses on only a small number of trees (typically one or two).

A. Forward Error Correction

To provide robustness to the system, FEC can be applied to the packets sent over different trees as depicted in Fig. 4. In this example, a subset of k trees is chosen to carry video packets and the remaining $n - k$ trees carry parity information. For each group of n packets, the different peers can decode the video stream, error-free, as long as k packets are received. Therefore, as long as a peer continues to receive data from at least k multicast trees, no error will be visible to the user. This system is particularly elegant, since video packets and/or parity information may also be regenerated at any intermediary peer as long as enough packets are received. This prevents the signal from degrading as it relayed down the multicast trees, as noted in [42].

The advantage of this scheme lies in its simplicity. Reed–Solomon encoding and decoding is an easy operation, especially since the number of packets, i.e., symbols, for this type of algorithm is very limited (typically on the order of ten). It is also flexible since the number of trees and the amount of parity information can be varied to obtain the desired robustness. On the other hand, the FEC procedure described in Fig. 4 does not explicitly provide an increasing level of protection to the different portions of a video stream according to their video importance. In addition, the $(n-k)/k$ overhead of this scheme can be large in comparison to the loss rate, which is typically low, on average, but highly bursty.

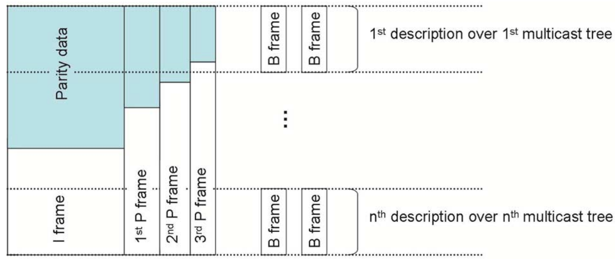

Fig. 5. ***Multiple descriptions are generated by applying different levels of protection to the different portions of the stream.***

B. Multiple Description Coding

Although multiple descriptions can be obtained in a variety of ways, in this paper we focus on the case where unequal error protection is applied to the different portions of a compressed video stream to produce a layered multiple description representation. This has the advantage of not requiring a special type of encoder to produce the stream, which would be necessary, if temporal or spatial subsampling were used to generate the different descriptions. The type of MD coding we describe can be obtained after a video stream has been compressed, for example, by widely available tools such as the VLC or QuickTime H.264/AVC encoders, or any other MPEG encoder.

To generate multiple descriptions, we separate the compressed video frames into different layers, according to their level of importance. For the encoding structure depicted in Fig. 2, I frames constitute the first layer, the GOP first P frames the second layer, the GOP second P frames the third layer, etc. The last layer is composed of the different B frames. An FEC code is applied to the different layers as illustrated in Fig. 5. The strength of the code, i.e., the rate expansion factor n/k, is chosen as linearly related to the video importance of the frames of the different layers. In the example shown in the figure and analyzed further in Section V, B frames are not protected. This assumes $n = k$ when the importance $\tilde{D} = 1$. Depending on the desired amount of total overhead, n/k can easily be determined for other layers, given their rate.

A variable number of descriptions is created by forming packets across the different layers as depicted in Fig. 5. Each description is associated with a different multicast tree. As B frames are not protected, they are simply transmitted by the source in round-robin order over the different trees. The number of layers that can be decoded depends on the number of descriptions received by the peers. In addition, the perceptually most important portions of the video stream are decodable even if only a small subset of descriptions is correctly transmitted. This makes for a very robust system, which could also be used to serve heterogeneous peers that do not have enough throughput to receive all the descriptions. Depending on the number of descriptions desired, each description might not lead to a perceptually significant signal if it is received alone. Therefore, this scheme could be labeled as a "generalized multiple description" scheme, although the information-theoretic definition of multiple descriptions [29] is much larger.

Similar to the FEC scheme, the independence of the different descriptions is well suited to transmission over multiple multicast trees, since losses on the different trees tend to be uncorrelated. In terms of rate, multiple description schemes share the same drawback as FEC: the total amount of overhead needed to protect the stream against burst errors may be large compared to the overall average packet loss rate. As a notable difference, since the different descriptions often combine parity information as well as video data, there is no simple way to regenerate a lost description when losses occur. Therefore, the signal quality may degrade as it is relayed down the multicast trees. Moreover, as the encoding occurs on a GOP basis, this procedure introduces additional delay, as the source cannot form the unequal error protection block before all the frames of the GOP have been compressed.

C. CoDiO P2P

Lastly, we present a multimedia scheduler, based on prioritized transmissions, feedback, and retransmissions, which has been the subject of our recent work. Different from network-level multicasting, the incorporation of application-level retransmission requests into P2P multicast is possible without feedback implosion, since the fanout of each individual node is small. Each peer node only serves a few other peers (and not hundreds or thousands of clients, as an IP-multicast media server). This permits fairly sophisticated scheduling algorithms.

Extending a server–client multimedia packet scheduler to P2P requires considering several new problems that characterize the particular nature of this transmission scenario. Unlike unicast, each peer receives video packets from a set of senders and forwards them to several receivers. This raises a number of interesting questions. How can a peer implement an adaptive forwarding transmission scheme and yet coordinate its scheduling policies with other senders forwarding video packets to the same descendant? Which of its different descendants should a peer favor when its resources are insufficient to serve all of them? The CoDiO P2P scheduler is a possible way to address these questions. This scheduler is composed of two parts. At the sender, a prioritization scheme determines the transmission order of video packets destined to multiple peers. At the receiver, a feedback mechanism is used to recover missing video packets when a peer is disconnected from one (or several) multicast tree(s).

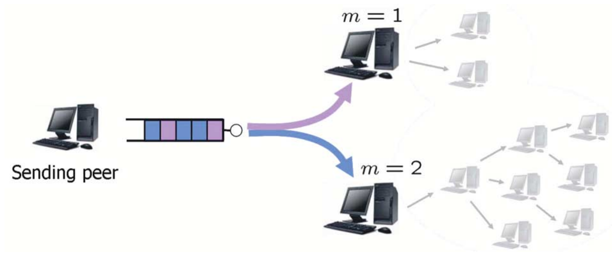

Fig. 6. ***The sending peer addresses packets of varying levels of importance to peers that subsequently forward them to varying numbers of descendants. Only one multicast tree is depicted in the figure for clarity.***

1) Sender-Driven Prioritization: Relaying traffic over the uplink of the peers may lead to congestion on the multihop path separating the source from any particular peer. In particular, because the rate of a video stream often varies or because of unexpected retransmission requests, a peer may sometimes lack the resources to forward all the data expected to its descendants. Optimized 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 video. The CoDiO P2P prioritization algorithm determines iteratively which is the next most important packet by comparing the video importance of each queued packet $\tilde{D}(n)$, defined in Section III.

The role of the scheduler is not only to determine in which order to send packets destined to a particular peer but also how to prioritize among the different descendants of a peer. Therefore, the importance of each packet should also be adjusted based on the number of descendants in the multicast tree that would be affected by the loss or late arrival of this packet, as depicted in Fig. 6. Hence, the scheduler should adapt its decisions to the video content *and* to the structure of the underlying multicast trees.

The CoDiO P2P scheduler bases its decisions on the unequal contribution of different packets to the overall video distortion. It also takes into account information collected about the structure of the multicast trees to favor peers with a large set of descendants. Its prioritization algorithm determines iteratively which is the next most important packet by comparing the *impact* of each queued packet. For a packet n, addressed to peer m, the impact is expressed as

$$I(n,m) = \tilde{D}(n) * (\text{NumDescendants}(m) + 1). \quad (1)$$

In (1), $\text{NumDescendants}(m)$ represents the number of peers to which packet n will be forwarded after reaching peer m; this information is collected by the control protocol when control packets are exchanged between neighboring peers periodically to maintain the multicast trees.

The prioritization is run as soon as more than one packet is in the transmission queue of the scheduler. Since the scheduler spaces its transmissions to avoid congesting the uplink, as described in the following, this occurs when the instantaneous transmission rate of the scheduler exceeds the throughput of the uplink of the peer.

The scheduler spaces successive transmissions to ensure congestion is not created on the bottleneck link of the network path. Transmissions are spaced based on the time needed for the previous video packet to traverse the uplink. In addition, a small fraction of the link capacity (20 kb/s in our simulations) is set aside. This is sufficient to ensure low congestion and to limit the delay of control packets. Please note that the rate of control traffic is between 2% and 4% of the total traffic exchanged on the network, as reported, for example, in [54] and [60]. Please also note that when a packet retransmission is requested by a descendant of the peer, the importance of the retransmission packet is computed according to (1). Therefore, the scheduler does not automatically favor retransmitted packets.

2) Distortion-Optimized Retransmission Scheduling: When a peer is disconnected from one or several multicast trees, a list of missing frames is determined by forming an estimated mapping between missing packets and their corresponding frame using the information contained in the header of other received video packets (packet number, frame number, number of packets in the frame, etc.). The scheduler uses its knowledge of the GOP structure to determine which missing frame has the highest contribution to the total expected distortion. The importance of missing frames is compared by computing the expected video quality associated with receiving each of them. Retransmission requests are sent out, packet by packet, in order of importance, according to this metric. When a peer is partially disconnected from a multicast session, retransmission requests are sent over the multicast trees a peer is still connected to, in round-robin order. Due to space limitations, we do not describe this algorithm in more detail; interested readers are referred to [56], where optimized retransmission scheduling is analyzed in depth.

It is also possible to consider a simpler retransmission scheme, based on feedback, where missing packets are requested sequentially. Our results show this is efficient as long as the delay constraint between the source and all the peers of the P2P system is sufficiently lax.

V. EXPERIMENTAL PERFORMANCE EVALUATION

A. Experimental Setup

To evaluate the performance of the different P2P video transport systems we describe in Section IV, we carry out experiments over a simulated network, in NS-2,[2] where all the peers run the SPPM protocol. This is convenient, as it allows repeatable experiments, which are essential to compare fairly different systems.[3] Simulations are run over a network topology with a few thousand nodes. The actual number of peers participating in each simulation varies between 1500 and 2000. The backbone links are sufficiently provisioned so that congestion only occurs on the links connecting the peers to the network. The delay over each link is 5 ms and the diameter of the network 10 hops. The peers, including the source, are randomly distributed at the edge of the network. The NS-2 files generating the topology and the connection patterns of the peers have been made available.[4] Losses are only due to disconnections or delay, and transmission errors due to the presence of Internet service provider boundaries or potential wireless last-hop links are ignored. The control and transmission protocol is implemented over the UDP/IP protocol stack, and we ignore any network address translator (NAT) or firewall issues that may limit connectivity.

Peers have heterogeneous but fixed uplink bandwidth, which they have measured and know accurately. The bandwidth of the peers reflects today's available asymmetric digital subscriber line network access technology. The bandwidth distribution is given in Table 1. It is derived from the findings of [44], which provides an estimate of the bandwidth of hosts connecting to media servers maintained by a leading content delivery network in 2003–2004. The uplink and downlink of the source are assumed to be 1.4 Mb/s.

In the experiments, the dynamic behavior of peers is modeled as follows. A flash crowd is simulated by letting 300 peers request the video during the first minute of the video session. During the remaining time, peers join and leave the session, following a random Poisson process. Peers remain connected for an average time of 4.5 min. After the initial transient period, the number of peers connected to the system is kept close to constant. The SPPM control protocol run by the peers is described in detail in [56], [57], and [61]. It operates in a distributed fashion and lets the peers build and maintain multiple multicast trees rooted at the source which carry complementary portions of the video stream. All the peers maintain connections to all the multicast trees. The protocol is designed to achieve short startup times, on the order of 1 s.

[2]www.isi.edu/nsnam/ns/.

[3]Results analyzing the deployment of CoDiO P2P over the SPPM protocol in PlanetLab were also reported, for example, in [60].

[4]http://www.ivms.stanford.edu/~esetton/tcl_files.htm.

Table 1 Distribution of Peer Bandwidth

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%

We show results for the common intermediate format (288×352) test video sequence *Foreman,* encoded at 30 frames/s with H.264/AVC at different rates. The encoding structure is that depicted in Fig. 2. This 10-s video clip is looped to obtain 30-min multicast sessions. Video quality is recorded at the different peers. It is measured in terms of average *peak signal-to-noise ratio (PSNR)*. The PSNR for a frame is a function of the mean squared error (MSE) between the decoded video frame and the original frame

$$\mathrm{PSNR} = 10\log_{10}\frac{(2^L-1)^2}{\mathrm{MSE}} \tag{2}$$

where L represents the number of bits used to encode the luminance signal, typically 8 bits. To avoid biases due to transient behavior, we exclude the first 100 s of the experiments when computing average video quality.

B. Performance Evaluation

We first analyze the performance of the different video transport schemes for P2P multicast sessions when a 290-kb/s video stream is sent over different numbers of multicast trees. Relative to the average throughput of the peers, this stream is not particularly difficult to accommodate. The playout deadline for all the peers is set to 2 s. For FEC, only one tree carries parity information, and the rest are used to forward video packets. Thus, the level of error resilience is determined by the number of trees, as both the robustness and the overhead decrease with the number of trees. For this scheme, the overhead is approximately $1/(n-1)$, where n is the number of trees. The size of parity packets is always equal to the largest packet in a block of packets, as depicted in Fig. 4. Therefore, to avoid creating unbalanced representations, the tree carrying parity information is alternated for each block of packets. To provide a meaningful comparison between MD and FEC, we match the amount of overhead of the two schemes in the different experiments. For CoDiO, the overhead is not computed a priori since this is an adaptive prioritized ARQ scheme. The overhead is determined by the number of lost packets that are requested by disconnected peers and by the uplink throughput of the peers that receive the retransmission requests. In addition to the FEC and MD schemes, and to the CoDiO scheduler, we also show the results of a simple reference transport scheme that does not incorporate any error resilience. For this scheme, video packets are sent over the different multicast trees in round-robin order, and no retransmission is allowed.

As shown in Figs. 7 and in 8, the performance of the schemes described in Section IV is quite close. The gap between the decoded video quality and the encoded quality is also small. Compared to the reference scheme where no error resilience is used at all, the percentage of frozen pictures is significantly reduced and video quality is increased by 1 dB, or more, for all schemes. For FEC and MD, the amount of robustness decreases with the number of trees, which explains the slight decrease in terms of video quality for a larger number of trees. On average the loss rate for the MD scheme is slightly above 2%. The frames with no or little protection such as B frames and the last P frames of a GOP make up for most losses. Hence, the resulting video quality is quite high, as can be seen in Fig. 8. As illustrated in Fig. 7, the FEC scheme is more robust; on average the percentage of lost frames is 1%. The only losses for FEC are due to disconnections, which affect more than one distribution tree at a time. In addition, FEC packets may be regenerated along the multicast trees as long as a sufficient number of packets in a block is received by the peers. This also contributes to a lower loss rate. However, this better performance does not translate into a large increase in terms of video quality, compared to MD, since FEC does not offer an increased protection to more perceptually important pictures like I frames. The adaptivity of the ARQ mechanism employed by CoDiO translates to even better performance. For this scheme, the average percentage of frozen frames is only 0.3%, three times less than for FEC and seven times less than for MD. This results in a marginally higher video quality. The results of additional experiments show that in this scenario, where transmission rate is limited, the gains largely come from adapting to the different packet types of the video stream.

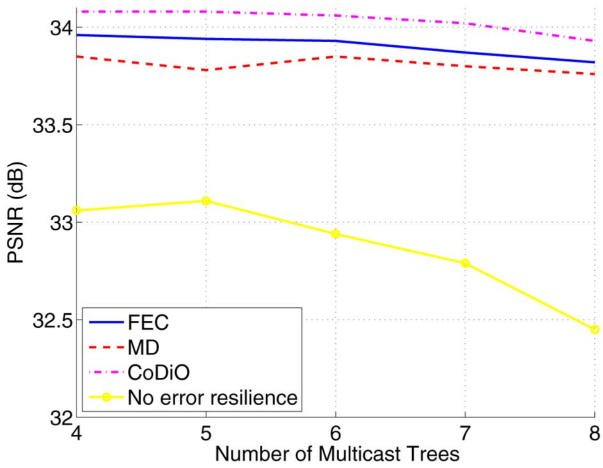

Fig. 8. ***Average video quality for all the participating peers. Results are shown for different transport schemes when different numbers of multicast trees are used to carry the video stream. The top of the y-axis represents the encoded video quality.***

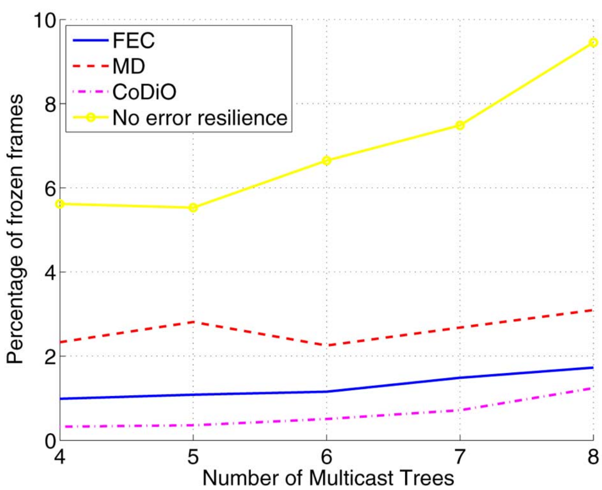

Fig. 7. ***Average residual loss rate for all the participating peers. Results are shown for different transport schemes when different numbers of multicast trees are used to carry the video stream.***

The results described above show the effectiveness of the three transport schemes when throughput on the P2P network is plentiful. In the following, we analyze the performance of the schemes when congestion is created on the network due to higher rates. To separate the effects due to the protocol and those due to the transport schemes, we study the situation where peers require 456 kb/s of free throughput to connect to the multicast session. This total available rate is split between the different trees built by the SPPM protocol. Since the average throughput available on the network (approximately 915 kb/s) exceeds this rate by a large amount, we know the resulting distribution trees will be stable. We analyze in Figs. 9 and 10 the effect of transmitting video at different rates along the multicast trees. In these experiments, the playout deadline is 2 s for all the peers. For FEC, we choose to transmit video over seven multicast trees, one of which is used to carry parity information. For our setup, this leads to the optimal tradeoff for this transport method between data overhead and error resilience at rates approaching the throughput of the multicast trees. In particular, it is for this number of trees that FEC can accommodate the highest video rate. Likewise, we fix the total amount of overhead for the MD encoding scheme to 30%, and the UEP block is divided into eight descriptions sent over eight trees. It is interesting to note that the optimal amount of overhead

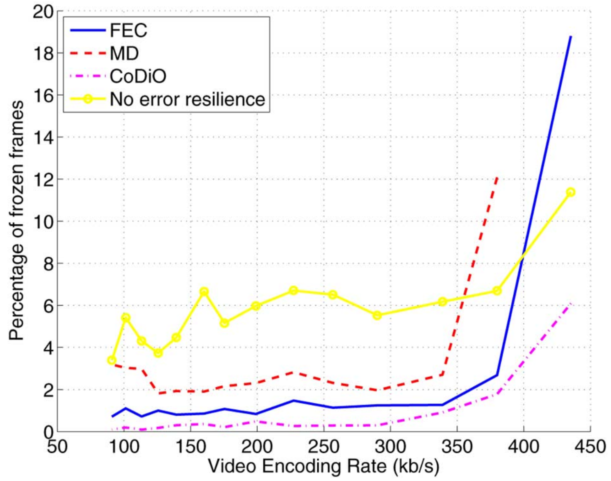

Fig. 9. ***Average residual loss rate for all the participating peers. Results are shown for different transport schemes as a function of the video encoding rate. Please note this rate does not include the channel coding overhead for FEC and MD, nor the retransmission rate for CoDiO.***

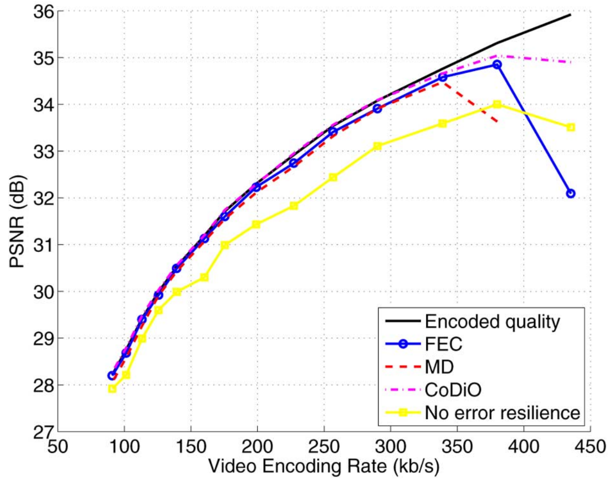

Fig. 10. ***Average decoded video quality for all the participating peers. Results are shown for different transport schemes as a function of the video encoding rate. Please note this rate does not include the channel coding overhead for FEC and MD, nor the retransmission rate for CoDiO.***

for both MD and FEC is very close.[5] For CoDiO, the best performance is obtained for five multicast trees.

The percentage of frozen frames is shown in Fig. 9. For rates below 350 kb/s, the three schemes perform reasonably well, compared to a reference scheme where no error resilience is used at all. The percentage of frozen pictures is significantly reduced, and video quality is significantly higher for all schemes. For these rates, the performance is similar to that observed for Figs. 7 and 8. The performance for higher rates indicates the tolerance of the schemes to self-inflicted congestion, when video traffic creates delay on the uplink of the peers. Both MD and FEC due to their larger amount of overhead create more congestion on the P2P network than CoDiO. Loss rates dramatically increase and video quality drops for all the schemes when they reach the capacity of the multicast trees. For MD and FEC, this occurs at lower video encoding rates, since these two schemes carry both the video packets and parity information. By comparison, the rate of retransmissions for CoDiO is very limited. Although MD and FEC transmit the same amount of overhead, on average, for a given stream, the performance of MD is worse than that of FEC at high rates. This is due to the lower tolerance of MD to congestion and is caused by the additional delay introduced when encoding the UEP block, on a GOP basis.

Lastly, we compare the limits of the three schemes in terms of end-to-end latency. Fig. 11 shows the average decoded video quality for the different peers as a function of the playout deadline. In this experiment, the sequence is encoded at 340 kb/s. The redundancy of the FEC scheme and of the MD scheme is comparable, and identical to that of the previous experiment. As illustrated, as long as the playout deadline is higher than 1.5 s, all the schemes achieve near optimal performance. For shorter playout deadlines, the decoded video quality decreases. This is due to congestion on the uplink of the peers, which causes excessive end-to-end delay, in particular for peers that lie towards the end of the different multicast trees. For the

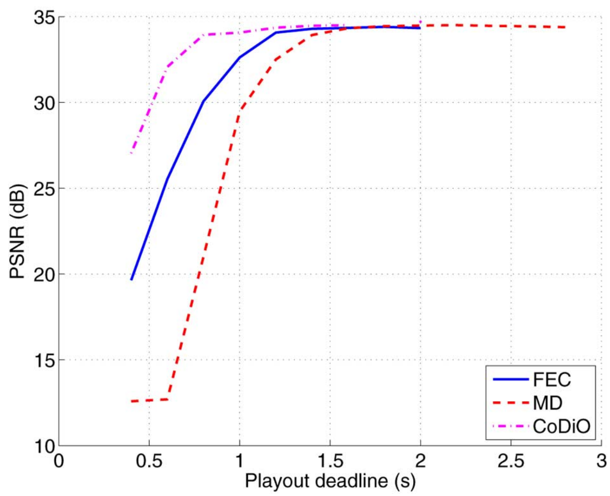

Fig. 11. ***Video quality for low-latency streaming. The performance of the three transport schemes is compared for different playout deadlines.***

[5]Although the theoretical fraction of overhead for FEC should be around one-sixth, since one tree out of seven is used to carry parity data, the actual rate is close to 30% due to the varying sizes of packets used to build the code in the process shown in Fig. 4.

MD scheme, the quality degradation occurs for longer playout deadlines, since the unequal error protection scheme described in Section IV introduces about half a second additional delay compared to the FEC scheme. CoDiO P2P can maintain high video quality for playout deadlines approximately 50% shorter than for MD and 33% shorter than for FEC. This is a significant difference for applications that target low latency. Two main factors contribute to these gains. Both FEC and MD transmit on average more traffic along the multicast tree, due to the overhead of the schemes. This causes additional congestion over the uplink throughput of the peers and results in delay. In addition, when the delay constraint is too stringent to transmit all the queued packets before their decoding deadlines, CoDiO P2P transmits in priority the most important packets to the peers which serve the largest number of subsequent peers. This results in improved video quality, despite the congestion-limited network. For CoDiO, adapting to the different packet types makes for a little more than half of the improvement compared to FEC, whereas the rest of the gain is due to adapting to the importance of the peers in the tree topology.

VI. DEPLOYMENT

We developed a real-time prototype of the SPPM that can run on both Linux and Windows platforms. Our current implementation constructs a set of multicast trees to forward video data encoded with the state-of-the-art H.264/AVC codec. The CoDiO P2P scheduler, described in Section IV, is used for error resilience and congestion control. We report performance measurements of the real-time SPPM system over PlanetLab in [60]. These results confirm the findings reported in previous sections. With a limited number of peers (approximately 100), startup latencies on the order of 1 s can be achieved. To the best of our knowledge, this is the lowest startup latency reported so far for a P2P streaming system. The development of a real-time system reveals numerous additional problems, which are orthogonal to the main focus of the paper. These different issues need to be addressed, however, to obtain a successful implementation. This process has also taught us how to evaluate a deployed P2P multicast client and how to compare the performance of different systems.

A. NATs and Firewalls

NATs are widely used over the Internet [10]. They associate a private IP address over a local network to different users sharing a public IP address. It is well known that NATs limit connectivity of P2P systems [62], [63]. As most other protocols, SPPM uses STUN [64] to access a rendezvous point that can provide public addresses to peers residing behind NATs.

Moreover, many network administrators try to limit the usage of P2P applications by disabling traffic forwarding across firewalls to protect local networks. To overcome this problem, P2P systems might have to use HTTP tunnelling to traverse a firewall. Such techniques usually increase the latency of the protocol. In addition, they impose the congestion control mechanism of transmission control protocol, which is not particularly well suited to low-latency video streaming.

B. Network Dynamics

Several P2P video streaming solutions use a set of multicast trees to deliver video data and rely on an estimate of the available throughput to construct an efficient overlay topology. For example, in SPPM, a peer will accept a new child on a particular multicast tree only if it has enough available throughput.

Since there is a significant dissymmetry between uplink and downlink throughput (see, e.g., Table 1), the uplink is often the bottleneck of the connection between a peer and its children. An accurate estimate of this throughput is essential to ensure that no unnecessary congestion is created over the P2P network and that the resources of the network are efficiently utilized. For example, in SPPM, the uplink throughput is continuously estimated by collecting statistics from the flow of received video packets. This information is provided to the algorithms responsible for maintaining the overlay topology. Congestion may be averted by monitoring that packet loss does not occur simultaneously between a peer and its different children. In addition, tracking end-to-end delay variation can also be used to infer throughput.

Another phenomenon that stresses P2P video streaming systems is peer churn. As a result of connections and disconnections, the overlay topology can deteriorate and end up in a brittle state. This problem is particularly obvious after massive disconnection events. In this case, a set of peers with low throughput can clog the system by connecting to the source of the stream. This can saturate the available bandwidth and starve other peers. In order to prevent such events, it is important to reconfigure the overlay network dynamically, thus continuously improving the distribution topology.

C. Framework for Comparing P2P Streaming Systems

Different from file-sharing applications, the performance of video streaming systems cannot be evaluated on the basis of purely network-based metrics like throughput, protocol overhead, or percentage of data received. In addition to these metrics, it is important to evaluate the impact of packet loss in terms of the actual video distortion observed by the users. Since real decoding of the signal over thousands of nodes can be computationally demanding, video traces are often used to estimate perceived video quality. Video quality estimation for Internet streaming using video traces has been extensively studied and documented—as, for example, in [65]—but without specific application to the context of P2P multicast. When losses occur, frame copy concealment can be simulated to evaluate the impact of error propagation. This is the approach we have taken in this paper. In this case, the loss of several consecutive frames creates a "frame freeze." The impact of such a loss pattern on video distortion may be predicted using a distortion table that indicates the effect of replacing a lost frame by a previous frame. Based on such a table, PSNR can be easily computed by comparing the expected decoded sequence with the original uncompressed signal.

Average PSNR over a very long sequence of frames is not always sufficient to accurately reflect video quality. Indeed, it could happen that connection failures cause video freezes for long periods of time while the observed drop in terms of average PSNR remains limited. Other metrics such as the frequency and length of frame freezes should be considered as an indicator of the frequency of failures of the P2P system. Finally, the initial delay required by the client before the first decodable frame can be displayed is another important metric to evaluate how suitable a particular P2P streaming system is for low-delay applications such as IP-TV.

VII. CONCLUSIONS

In this paper, we give an overview of error-resilient transport for P2P multicast systems. The different schemes we consider vary from a pure channel coding technique to a cross-layer system specifically designed for video streams and multicast distribution trees. Our analysis is motivated by the unique properties of compressed video signals. Live video streams are composed of delay-constrained packets with vastly varying perceptual importance; they call for algorithms that adapt to, and take advantage of, these differences. We report the following findings from our experimental results.

- Due to peer churn, P2P video streaming systems need to incorporate a high degree of robustness to achieve good performance.
- As long as the latency and throughput requirements of the system are not too stringent, generic error-resilient schemes and cross-layered designed schemes that adapt to the properties of video traffic perform comparably well.
- Cross-layered techniques such as the congestion-distortion optimized system described in this paper, where packet scheduling adapts to both the underlying P2P topology and to the video stream, are particularly effective for streaming with very low latencies or high rates.

The growing popularity of video streaming over P2P networks makes it an exciting research topic, particularly since adaptive video streaming in these environments is still in its early stages. Many algorithms designed for server–client streaming systems can, and should, be adapted to P2P and will lead to interesting results, which will be important for next-generation P2P streaming systems. ■

Acknowledgment

The authors would like to thank the members of the Image, Video, and Multimedia Systems Group, Stanford University, for many stimulating discussions, as well as the Streaming Media Systems Group, and in particular Dr. J. Apostolopoulos, at Hewlett-Packard Laboratories, for continued encouragement and support of their work.

REFERENCES

[1] J. Liu, S. Rao, B. Li, and H. Zhang, "Opportunities and challenges of P2P internet broadcast," *Proc. IEEE*, vol. 96, no. 1, pp. XXX–XXX, Jan. 2008.

[2] W. Zhu, M.-T. Sun, L.-G. Chen, and T. Sikora, Eds., "Special issue on advances in video coding and delivery," *Proc. IEEE*, vol. 93, Jan. 2005.

[3] W. Zeng, K. Nahrstedt, P. Chou, A. Ortega, P. Frossard, and H. Yu, Eds., "Special issue on streaming media," *IEEE Trans. Multimedia*, vol. 6, Apr. 2004.

[4] B. Girod, M. Kalman, Y. Liang, and R. Zhang, "Advances in channel-adaptive video streaming," *Wireless Commun. Mobile Comput.*, vol. 6, no. 2, pp. 549–552, Sep. 2002.

[5] M. Civanlar, A. Luthra, S. Wenger, and W. Zhu, Eds., "Special issue on streaming video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, Mar. 2001.

[6] Y. Chu, S. Rao, S. Seshan, and H. Zhang, "A case for end system multicast," *IEEE J. Sel. Areas Commun.*, vol. 20, pp. 1456–1471, Oct. 2002.

[7] A. Ganjam and H. Zhang, "Internet multicast video delivery," *Proc. IEEE*, vol. 93, pp. 159–170, Jan. 2005.

[8] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast," in *Proc. ACM SIGCOMM*, Pittsburgh, PA, Aug. 2002, pp. 205–217.

[9] D. Tran, K. Hua, and T. Do, "ZIGZAG: An efficient peer-to-peer scheme for media streaming," in *Proc. IEEE INFOCOM*, San Francisco, CA, Mar. 2003, vol. 2, pp. 1283–1292.

[10] 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," in *Proc. USENIX Annu. Tech. Conf.*, Boston, MA, Jun. 2004, pp. 1283–1292.

[11] Y. Zhu, B. Li, and J. Guo, "Multicast with network coding in application-layer overlay networks," *IEEE J. Sel. Areas Commun.*, no. 22, pp. 107–120, Jan. 2004.

[12] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum, "DONet/coolStreaming: A data-driven overlay network for live media streaming," in *Proc. IEEE Infocom*, Miami, FL, Feb. 2005.

[13] M. Zhang, J.-G. Luo, L. Zhao, and S.-Q. Yang, "A peer-to-peer network for live media streaming using a push-pull approach," in *Proc. ACM Int. Conf. Multimedia*, Nov. 2005, pp. 287–290.

[14] M. Zhang, J.-G. Luo, L. Zhao, and S.-Q. Yang, "Large-scale live media streaming over peer-to-peer networks through global internet," in *Proc. ACM Int. Conf. Multimedia (P2PMMS Workshop)*, Nov. 2005, pp. 21–28.

[15] A. Albanese, J. Blömer, J. Edmonds, M. Luby, and M. Sudan, "Priority encoding transmission," *IEEE Trans. Inf. Theory*, vol. 42, pp. 1737–1744, Nov. 1996.

[16] U. Horn, K. Stuhlmüller, M. Link, and B. Girod, "Robust internet video transmission based on scalable coding and unequal error protection," *Signal Process. Image Commun.*, vol. 15, no. 12, pp. 77–94, Sep. 1999.

[17] A. Mohr, E. Riskin, and R. Ladner, "Unequal loss protection: Graceful degradation of image quality over packet erasure channels through forward error correction," *IEEE J. Sel. Areas Commun.*, vol. 18, pp. 819–829, Jun. 2000.

[18] W. Zhu, Q. Zhang, and Y.-Q. Zhang, "Network-adaptive rate control with unequal loss protection for scalable video over internet," in *Proc. Int. Symp. Circuits Syst.*, Sydney, Australia, May 2001.

[19] R. Puri and K. Ramchandran, "Multiple description source coding through forward error correction codes," in *Proc. IEEE Asilomar Conf. Signals, Syst., Comput.*, Asilomar, CA, Oct. 1999, vol. 1, pp. 342–246.

[20] D. Turner and K. Ross, "Optimal streaming of layered-encoded multimedia presentations,"

in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, New York, Jul. 2000.

[21] T. Tian, A. Li, J. Wen, and J. Villasenor, "Prority dropping in network transmission of scalable video," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Vancouver, BC, Canada, Oct. 2000, vol. 3, pp. 400–403.

[22] S. Dumitrescu, Z. Wang, and X. Wu, "Globally optimal uneven error-protected packetization of scalable code streams," in *Proc. Data Compression Conf.*, Snowbird, UT, Sep. 2002, pp. 73–82.

[23] J. Boyce, "Packet loss resilient transmission of MPEG video over the Internet," *Signal Process. Image Commun.*, vol. 15, no. 1–2, pp. 7–24, Sep. 1999.

[24] M. Hannuksela, Y.-K. Wang, and M. Gabbouj, "Isolated regions in video coding," *IEEE Trans. Multimedia*, vol. 6, pp. 259–267, Apr. 2004.

[25] P. Baccichet, S. Rane, A. Chimienti, and B. Girod, "Robust low-delay video transmission using H.264/AVC redundant slices and flexible macroblock ordering," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, San Antonio, TX, Oct. 2007.

[26] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: High-bandwidth multicast in cooperative environments," in *Proc. ACM Symp. Oper. Syst. Principles (SOSP)*, Oct. 2003.

[27] M. Bansal and A. Zakhor, "Path diversity for overlay multicast streaming," in *Proc. Packet Video Worshop*, Irvine, CA, Dec. 2004.

[28] Y. Liang, J. Apostolopoulos, and B. Girod, "Analysis of packet loss for compressed video: Does burst-length matter?" in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Hong-Kong, China, Apr. 2003, pp. 684–687.

[29] V. Goyal, "Multiple description coding: Compression meets the network," *IEEE Signal Process. Mag.*, vol. 18, pp. 74–93, Sep. 2001.

[30] J. G. Apostolopoulos, "Reliable video communication over lossy packet networks using multiple state encoding and path diversity," in *Proc. SPIE Conf. Visual Commun. Image Process. (VCIP)*, San Jose, CA, Jan. 2001, pp. 392–409.

[31] Y. Wang, S. Panwar, S. Lin, and S. Mao, "Wireless video transport using path diversity: Multiple description vs. layered coding," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Rochester, NY, Sep. 2002, pp. 21–24.

[32] A. Majumdar, R. Puri, and K. Ramchandran, "Distributed multimedia transmission from multiple servers," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Rochester, NY, Sep. 2002, pp. 177–180.

[33] S. Mao, S. Lin, S. Panwar, Y. Wang, and E. Celebi, "Video transport over ad hoc networks: Multistream coding with multipath transport," *IEEE J. Sel. Areas Commun.*, vol. 21, pp. 1721–1737, Dec. 2003.

[34] A. Begen, Y. Altunbasak, O. Ergun, and M. Ammar, "Multipath selection for multiple description video streaming over overlay networks," *Signal Process. Image Commun.*, vol. 20, no. 1, pp. 39–60, Jan. 2005.

[35] J. Apostolopoulos, "Error resilient video compression via multiple state streams," in *Proc. Int. Workshop Very Low Bitrate Video Coding (VLBV'99)*, Kyoto, Japan, Oct. 1999, pp. 168–171.

[36] S. Ekmekci and T. Sikora, "Unbalanced quantized multi-state video coding: Potentials," in *Proc. IEEE Picture Coding Symp.*, San Francisco, CA, Dec. 2004.

[37] V. Goyal and J. Kovacevic, "Generalized multiple description coding with correlated transforms," *IEEE Trans. Inf. Theory*, vol. 47, no. 6, pp. 2199–2224, Sep. 2001.

[38] I. Bajic and J. Woods, "Domain-based multiple description coding of images and video," in *Proc. SPIE Conf. Visual Commun. Image Process. (VCIP)*, San Jose, CA, Jan. 2002, pp. 124–135.

[39] S. Lin and Y. Wang, "Analysis and improvement of multiple description motion compensation video coding for lossy packet networks," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Rochester, NY, Sep. 2002, pp. 185–188.

[40] T. Petrisor, C. Tillier, B. Pesquet-Popescu, and J.-C. Pesquet, "Redundant multiresolution analysis for multiple description video coding," in *Proc. IEEE Int. Workshop Multimedia Signal Process. (MMSP)*, Siena, Italy, Sep. 2004.

[41] I. Radulovic and P. Frossard, "Multiple description image coding with block-coherent redundant dictionaries," in *Proc. Picture Coding Symp.*, Hangzhou, China, Apr. 2006.

[42] V. Padmanabhan, H. Wang, and P. Chou, "Resilient peer-to-peer streaming," in *Proc. IEEE Int. Conf. Network Protocols*, Atlanta, GA, Nov. 2003.

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

[44] K. Sripanidkulchai, A. Ganjam, B. Maggs, and H. Zhang, "The feasibility of supporting LargeScale live streaming applications with dynamic application EndPoints," in *Proc. SIGCOMM'04*, Portland, USA, Aug. 2004.

[45] S. Wicker, *Error Control Systems for Digital Communication and Storage*. Englewood Cliffs, NJ: Prentice-Hall, 1995.

[46] M. Podolsky, S. McCanne, and M. Vetterli, Soft ARQ for layered streaming media, Computer Science Division, Univ. of California, Berkeley, Tech. Rep. UCB/CSD-98-1024, Nov. 1998.

[47] Z. Miao and A. Ortega, "Optimal scheduling for streaming of scalable media," in *Proc. IEEE Asilomar Conf. Signals, Systems, and Computers*, Pacific Grove, CA, Nov. 2000, vol. 2, pp. 1357–1362.

[48] P. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," *IEEE Trans. Multimedia*, vol. 8, pp. 390–404, Apr. 2006.

[49] M. Kalman, P. Ramanathan, and B. Girod, "Rate-distortion optimized streaming with multiple deadlines," in *Proc. Int. Conf. Image Process. (ICIP)*, Barcelona, Spain, Sep. 2003, vol. 5, pp. 3145–3148.

[50] M. Röder, J. Cardinal, and R. Hamzaoui, "On the complexity of rate-distortion optimal streaming of packetized media," in *Proc. Data Compression Conf.*, Snowbird, UT, Mar. 2004.

[51] E. Setton and B. Girod, "Congestion-distortion optimized scheduling of video," in *Proc. Multimedia Signal Process. Workshop (MMSP)*, Siena, Italy, Oct. 2004, pp. 99–102.

[52] E. Setton, X. Zhu, and B. Girod, "Congestion-optimized scheduling of video over wireless ad hoc networks," in *Proc. Int. Symp. Circuits Syst. (ISCAS)*, Kobe, Japan, May 2005, vol. 4, pp. 3531–3534.

[53] E. Setton, "Congestion-aware video streaming over peer-to-peer networks," Ph.D. dissertation, Stanford Univ., Stanford, CA, 2006.

[54] E. Setton and B. Girod, *Peer-to-Peer Video Streaming*. New York: Springer, 2007.

[55] D. Anderson and J. Kubiatowicz, "The worldwide computer," *Sci. Amer.*, vol. 286, no. 3, pp. 28–35, Mar. 2002.

[56] E. Setton, J. Noh, and B. Girod, "Rate-distortion optimized video peer-to-peer multicast streaming," in *Proc. ACM Workshop Adv. Peer-to-Peer Multimedia Streaming (P2PMMS'05)*, Singapore, Nov. 2005, pp. 39–45.

[57] E. Setton, J. Noh, and B. Girod, "Congestion-distortion optimized peer-to-peer video streaming," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Atlanta, GA, Oct. 2006.

[58] Advanced Video Coding for Generic Audiovisual Services, ITU-T Recommendation H.264-ISO/IEC 14496-10(AVC), 2003.

[59] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, *RTP: A Transport Protocol for Real-Time Applications*, RFC 1889, Jan. 1996.

[60] P. Baccichet, J. Noh, E. Setton, and B. Girod, "Content-aware P2P video streaming with low latency," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Beijing, China, Jul. 2007.

[61] E. Setton, J. Noh, and B. Girod, "Low latency video sreaming over peer-to-peer networks," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Toronto, ON, Canada, Jul. 2006, pp. 569–572.

[62] B. Ford, P. Sisuresh, and D. Kegel, "Peer-to-peer communication across network address translators," in *Proc. USENIX Annual Tech. Conf.*, Anaheim, CA, Apr. 2005, pp. 179–192.

[63] A. Biggadike, D. Ferullo, G. Wilson, and A. Perrig, "NATBLASTER: Establishing TCP connections between hosts behind NATs," in *Proc. ACM SIGCOMM Asia Workshop*, Beijing, China, Apr. 2005.

[64] J. Rosenberg, J. Weinberger, C. Huitema, and R. Mahy, *STUN—Simple traversal of user datagram protocol (UDP) through network address translators (NATs)*, RFC 3489, Mar. 2003

[65] P. Seeling, F. Fitzek, and M. Reisslein, *Video Traces for Network Performance Evaluation*. New York: Springer, 2007.

ABOUT THE AUTHORS

Eric Setton (Member, IEEE) received the B.S. degree from Ecole Polytechnique, Palaiseau, France, in 2001 and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, CA, in 2003 and 2006, respectively.

He is the Chief Technology Officer and co-founder of Dyyno Incorporated, a company which provides a novel multimedia communication solution. Prior to joining Dyyno, he was a Research Scientist with the Streaming Media Systems Group at Hewlett-Packard Laboratories. He spent some time working at Sony, in 2003, and Sagem SA, in 2001. He is the author or coauthor of more than 25 publications in his field and of one book. He also has eight patents either granted or pending.

Dr. Setton received the Carnot Fellowship and the SAP Stanford Graduate fellowship in 2001 and the Sony SNRC Fellowship in 2003. His research interests range from multimedia communication over peer-to-peer networks to streaming over wired and wireless networks and video compression.

Pierpaolo Baccichet (Member, IEEE) received the M.S. and Ph.D. degrees in computer science from the University of Milan, Italy, in 2001 and 2006, respectively.

During his doctoral studies, he worked on low-complexity protection schemes for error-resilient video transmission at the National Research Center of Italy, Turin, funded by ST Microelectronics. He is currently a Visiting Assistant Professor with the Max Planck Center for Visual Computing and Communication, Stanford University, Stanford, CA. His research interests include error-resilient video coding and real-time streaming over peer-to-peer networks.

Bernd Girod (Fellow, IEEE) received the M.S. degree from Georgia Institute of Technology, Atlanta, and the doctoral degree in engineering from the University of Hannover, Germany.

He has been a Professor of electrical engineering and (by courtesy) Computer Science in the Information Systems Laboratory, Stanford University, Stanford, CA, since 1999. Previously, he was a Professor of telecommunications with the Electrical Engineering Department, University of Erlangen-Nuremberg, Germany. His current research interests are in the areas of video compression and networked media systems. He has published more than 400 conference and journal papers as well as five books. He has been involved with several startup ventures as Founder, Director, Investor, or Advisor, among them Polycom (Nasdaq: PLCM), Vivo Software, 8×8 (Nasdaq: EGHT), and RealNetworks (Nasdaq: RNWK). Since 2004, he has been Chairman of the new Deutsche Telekom Laboratories, Berlin, Germany.

Prof. Girod is a member of the German Academy of Sciences (Leopoldina). He received the EURASIP Best Paper Award in 2002 and the IEEE Multimedia Communication Best Paper Award in 2007, as well as the EURASIP Technical Achievement Award in 2004.