Enabling Adaptive Video Streaming in P2P Systems

Dan Jurca, Jacob Chakareski, Jean-Paul Wagner, and Pascal Frossard,
Ecole Polytechnique Federale de Lausanne (EPFL)

This work was supported by the Swiss National Science Foundation under grant PP-002-68737 and the Swiss Innovation Promotion Agency, under grant CTI - 7388.2 ESPP-ES.

ABSTRACT

Peer-to-peer (P2P) systems are becoming increasingly popular due to their ability to deliver large amounts of data at a reduced deployment cost. In addition to fostering the development of novel media applications, P2P systems also represent an interesting alternative paradigm for media streaming applications that can benefit from the inherent self organization and resource scalability available in such environments. This article presents an overview of application and network layer mechanisms that enable successful streaming frameworks in peer-to-peer systems. We describe media delivery architectures that can be deployed over P2P networks to address the specific requirements of streaming applications. In particular, we show how video-streaming applications can benefit from the diversity offered by P2P systems and implement distributed-streaming and scheduling solutions with multi-path packet transmission.

INTRODUCTION

P2P (peer-to-peer) networking architectures receive a lot of attention today because they enable a variety of new applications that can take advantage of the distributed storage and increased computing resources offered by such networks. In addition, P2P systems represent a scalable and cost-effective alternative to classic media delivery services that enables extended network coverage in the absence of IP multicast or expensive content distribution networks (CDN). The advantage of P2P systems resides in their capability for self organization, bandwidth scalability, and network path redundancy, which are all very attractive features for effective delivery of media streams over networks.

However, some fundamental differences between centralized/structured architectures and P2P systems first must be addressed to provide efficient P2P streaming solutions to existing media applications. Typical client-server architectures and CDN offer the network infrastructure that permits deployment of generic media applications (Fig. 1). The network infrastructure facilitates implementation of tools for effective rich media delivery, for example, end-to-end error correction, path computation, route selection, and rate adaptation. These tools generally rely on the centralized paradigm that is served by the sustained computational capabilities of streaming servers or cooperating proxy servers. P2P systems are less reliable but present the advantage of low-cost service deployment and the flexibility of resource aggregation through multiple path transmission (Fig. 2).

Researchers proposed to exploit the advantages of path diversity and system scalability in P2P environments to build some early streaming mechanisms, such as CoolStreaming [1]. However, the specificity of media applications in terms of bandwidth, delay, and reliability are not completely addressed by the characteristics of unstructured P2P systems. The lack of coordination of such systems, the limited peer capabilities, and the low system stability over time represent a great challenge for the deployment of high quality P2P streaming applications. The replacement or extension of conventional media delivery infrastructures with P2P systems clearly requires the adaptation of existing coding, routing, and scheduling algorithms to unreliable network environments.

The aim of this article is to describe state-of-the-art strategies that enable deployment of efficient streaming solutions in P2P systems. We present delivery architectures and adaptive streaming mechanisms that enable resource-demanding and delay-constrained applications over unstructured networks. We later review in more detail the strategies available for multi-path video transmission, the objective of which is to provide high sustainable bandwidth to the streaming applications. The networking mechanisms that achieve efficient, distributed packet scheduling and forwarding in P2P media systems also are examined. We eventually show that the P2P paradigm, along with adaptive streaming mechanisms, provides an interesting alternative for low-cost and effective multimedia communication applications.

0163-6804/07/$20.00 © 2007 IEEE

Peer-to-Peer Streaming Framework

Delivery Architectures

As a P2P system does not provide any guaranteed support to streaming services, these must rely on self-organized and adaptive network architectures to meet their stringent quality requirements. Two main types of architectures generally are considered for providing the organization required for streaming applications: tree-based overlay for streaming sessions from media sources to a pool of client peers and mesh overlay for massive parallel content distribution among peers.

On the one hand, tree-based overlays organize the peers as a single or multiple tree overlay that connects the source of the media content to the clients (Fig. 3). Clients are leaf nodes in the distribution tree, while intermediate peers push the content from the source. A peer can simultaneously be a leaf in some distribution trees and an intermediate node in others. Single tree architectures are easy to implement and maintain, either in a distributed or centralized way, by the source. However, they are fundamentally limited by the following two factors:

- Due to the high rate of peers joining/leaving the system (the so called churn rate), the architecture suffers from high instability;
- The received media quality is limited by the minimum upload bandwidth of the intermediate peers in the branch, as each client is connected to the source through a single tree branch.

Multiple tree architectures address the aforementioned problems, by providing redundancy in network paths. However, designing and maintaining such systems becomes less trivial. These also can lead to solving contradictory issues such as minimizing tree depth, while simultaneously provisioning network path diversity. Most importantly, the underlying physical topology must be carefully considered to achieve efficient content dissemination [2].

On the other hand, a mesh overlay architecture is based on self organization of nodes in a directed mesh that is used for media delivery to clients (Fig. 4). The original media content from a source is distributed among different peers. A peer is connected to the mesh through one or more parent peers, where it retrieves media information and a set of child peers, to which it serves media packets. The advantages of such an architecture reside in the low cost and simplicity of structural maintenance and in the resilience of the topology to node failure or departure, due to the increased probability of available distinct network paths. However, streaming applications over such architectures faces important challenges. First, due to the inherent sequential media encoding and play-out, packet dissemination and data requests must follow closely the temporal ordering of the content at the source. This constraint may be slightly reduced by the implementation of play-back buffers when delays permit it. Second, the limited look-ahead content availability, especially in the case of live streaming scenarios, greatly limits the flexibility in terms of content download/upload through such an architecture [3].

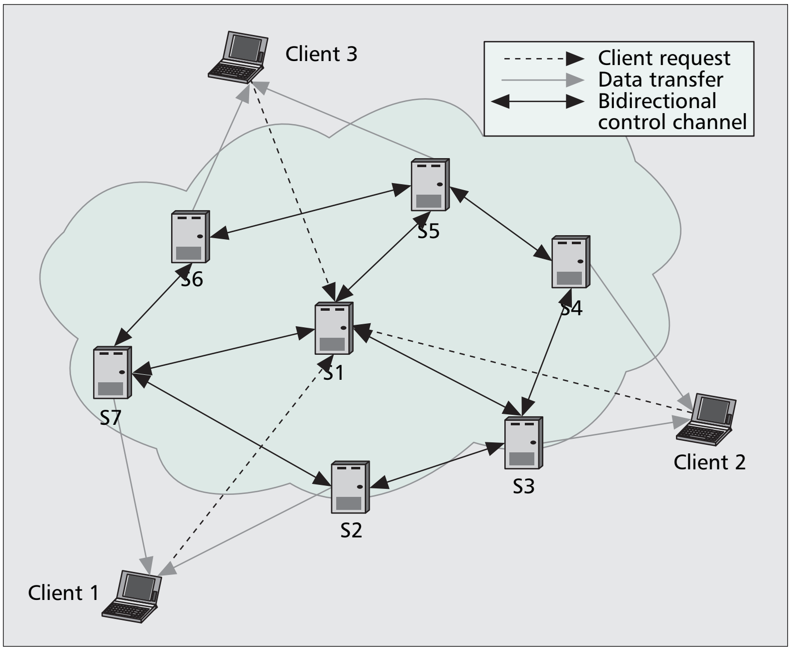


Figure 1. *Structured content distribution network: The client requests a service from the main application server; the servers cooperate among each other and decide which server or proxy has to serve the client.*

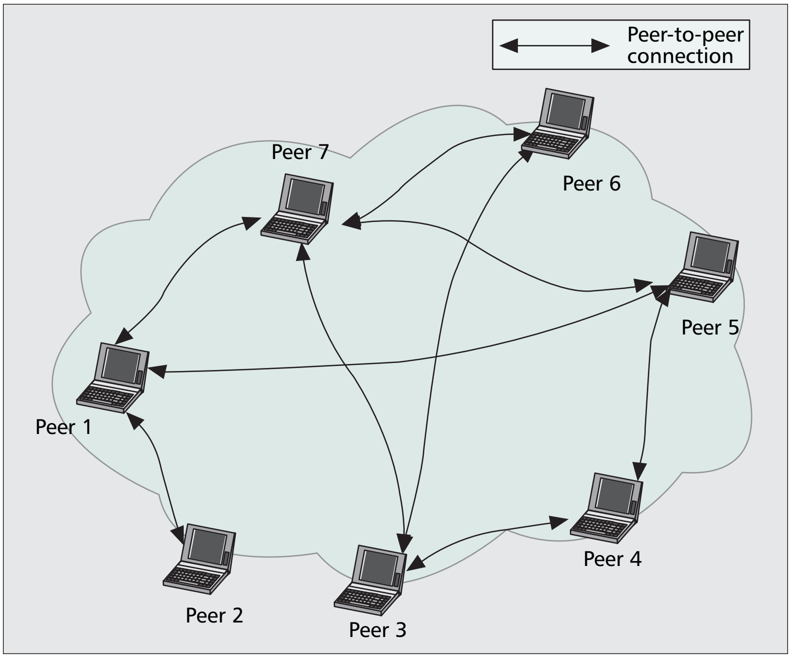
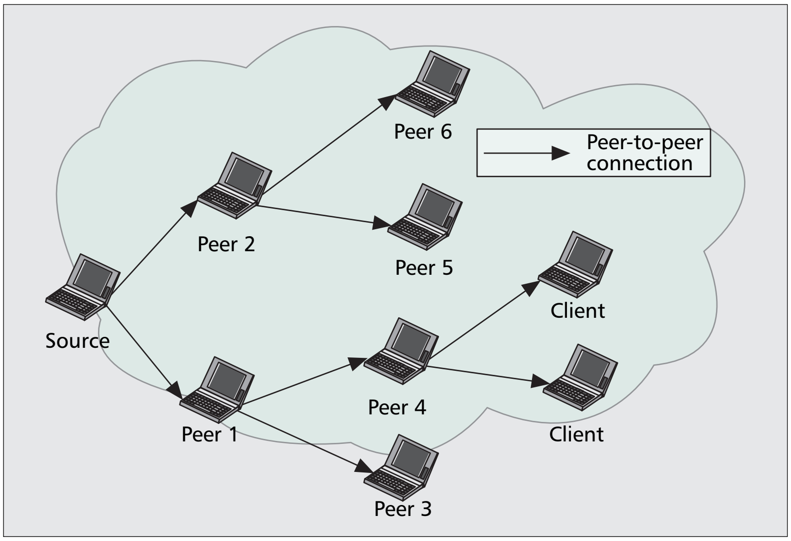


Figure 2. *Unstructured peer-to-peer network: the receiving peers connect and retrieve data from other peers.*

The design of efficient media-streaming solutions over each of these architectures requires adaptive and robust streaming strategies to overcome the variability and unreliability of the underlying transport medium. Media-specific solutions that allow for stream adaptation to the constraints and the specificity of the network are detailed in the next section. Effective routing and scheduling mechanisms that rely on increased network diversity are presented subsequently.



■ **Figure 3.** *Tree architecture for media delivery in P2P systems.*

MEDIA CODING FOR UNSTABLE P2P SYSTEMS

The lack of any quality of service (QoS) guarantee in typical P2P systems is in sharp contrast to the strict timing requirements of video streaming applications. In particular, the time-constrained video packets that must be delivered over networks are characterized by dynamic variations in bandwidth, loss rate, and delay jitter. The task becomes even more challenging due to the typically high and time-varying data rate of compressed video sources. Finally, peers usually join and depart the network at random, which represents yet another degree of difficulty for successful deployment of streaming applications over such networks. Media adaptation strategies at the application layer can be employed to address these challenges. Specifically, the video information can be compressed and packaged in a form that facilitates adaptation to variable network bandwidth, packet loss, and delay. Increased robustness of streaming applications can be achieved by introducing error control in the form of efficient packet retransmission and forward error correction.

The rate profile of a compressed video stream is typically independent of the network bandwidth variations. Therefore, rate adaptation of the video source must be performed to address the eventual mismatch between the two. Scalable video encoding provides an elegant solution. In particular, at compression, a scalable representation of the video source is created such that it enables scalability of the video stream in terms of its temporal, spatial, and SNR (signal-to-noise ratio or video quality) resolution. Scalability properties provide the capability to meet the constraints imposed by the bandwidth available at a given point in a P2P network. The scalable video content typically is organized in a hierarchy of layers, where the higher layers in each scalability dimension (space, time, and SNR) are discarded first in the event of insufficient network bandwidth, via in-network packet filtering and rate adaptation. As the same compressed content can be used to serve a variety of receiving clients, scalable representations are particularly advantageous for overlay architectures where there is a large heterogeneity between the nodes (peers) in terms of their access bandwidth and processing power.

An alternative technique to scalable coding for streaming applications is multiple description coding (MDC), which consists in constructing several independent descriptions of the same signal. In this approach, a controlled level of redundancy is left in the media content at compression so that the received video quality is proportional to the number of descriptions that are received. MDC represents a natural solution for multipath streaming scenarios, where independent descriptions can be sent on disjoint paths. It is generally less efficient than scalable encoding in terms of compression; however, it exhibits stronger resilience to packet loss. Two independent descriptions of the video content can be created by treating the odd and even frames of the video separately and can be sent over two separate network paths. Another approach to creating multiple descriptions is based, for example, on the application of forward error correction (FEC). In particular, different levels of redundancy in terms of FEC packets are applied to a set of media packets so that each redundancy level effectively corresponds to one independent description. The advantages and drawbacks of multiple descriptions and scalable coding for video streaming over overlay networks were studied most recently in [4].

Losses of media packets can be attributed to random peer departures, events of network congestion, and transmission over unreliable channels (e.g., wireless links). The media content can be compressed such that it exhibits a higher level of resilience to packet loss. In other words, the reconstructed video becomes less susceptible to error propagation that naturally occurs in predictive video coding when not all of the video packets are received (on time). Some of these tools, such as flexible macroblock ordering, multi-reference frame motion estimation, and redundant slices were incorporated in the most recent video coding standard H.264. Alternatively, missing packets can be recovered at the receiver via application layer retransmissions (ARQ) if the timing constraints of the streaming application permit it. However, when retransmission is unfeasible due to the related latency, FEC-redundant packets can be sent together with the video packets so that missing packets at the receiver can be recovered through FEC decoding. In particular, the FEC packets can be organized such that they provide unequal error protection (UEP) to scalable media representation [5]. Finally, receiver techniques, for example, error concealment or adaptive playout, can be enabled to mitigate the effects of eventual packet loss and provide an additional improvement in video quality.

The previous adaptive techniques can be elegantly combined with the availability of multiple sending sources or multiple network paths between a sender and a destination receiver. This so-called path diversity property arises naturally in the context of P2P overlay networks.

Routing and Rate Allocation in P2P Networks

Multi-path Streaming in Mesh Networks

Effective streaming mechanisms exploit the multi-path nature of P2P networks to satisfy the bandwidth requirements of media applications by aggregation of network resources. The early work presented in [6] establishes a generic framework for multi-path streaming. Some of the specific advantages brought by the utilization of multiple transmission paths for media dissemination consist of aggregated network bandwidth, packet loss de-correlation, and delay reduction. Prior experimental work on multipath streaming [7] offers some insights concerning the selection of content sources and streaming paths, based on the jointness/disjointness of network segments. However, these findings cannot be applied directly in P2P scenarios, especially due to the lack of coordination among the peers.

In a large network set up, when a client can connect to multiple source peers through distinct network paths, the streaming application must determine the best subset of paths and possible sources, along with the optimal rate allocation, on the chosen paths. This selection is based on network parameters, such as available path bandwidth and error rates and on media specific parameters. The media quality at the receiving peer can be maximized by proper path selection and rate allocation, which become application-specific to reflect the client satisfaction. Solutions to these problems lie in the cooperation between the media application and the path selection mechanisms. The encoding flexibility provided by the media-specific tools influences the choices made by the application at the transport and routing levels and provides the required adaptation to the changing network environment. However, it is not trivial to determine, in a distributed way, the optimal source selection and media rate allocation. Fluctuating topologies that are typical of P2P systems require periodic path re-computation and adaptation of the media application to cope with the channel and path variations. Next, we present, in more detail, receiver-driven and distributed strategies for effective routing and rate allocation in P2P networks.

Receiver-Driven Streaming Scenarios

Source peer selection and rate allocation typically are addressed in receiver-driven streaming scenarios, where the client coordinates the streaming process. Note that such a scenario, even if not fully decentralized, is a good strategy for P2P streaming systems. Content location information can be accessed by the receiver at super nodes/servers as in BitTorrent or PPLive solutions or from other peers by search algorithms adapted to decentralized systems. Furthermore, the receiving peer can probe for network connections toward candidate source nodes. Then, based on network connectivity information and streaming session characteristics, a receiver makes an informed choice of source peers and network transmission paths [8]. Furthermore, during a streaming session, a receiver has access to network path statistics (e.g., via RTCP reports). Hence, it is able to aggregate this information and construct a timely image of the available network topology. Finally, application adaptation can be performed by the receiver, to reflect the changes observed in the transport medium. Not surprisingly, prior work [9] advocates the choice of lowest error paths first for media delivery from multiple source peers to a receiving end.

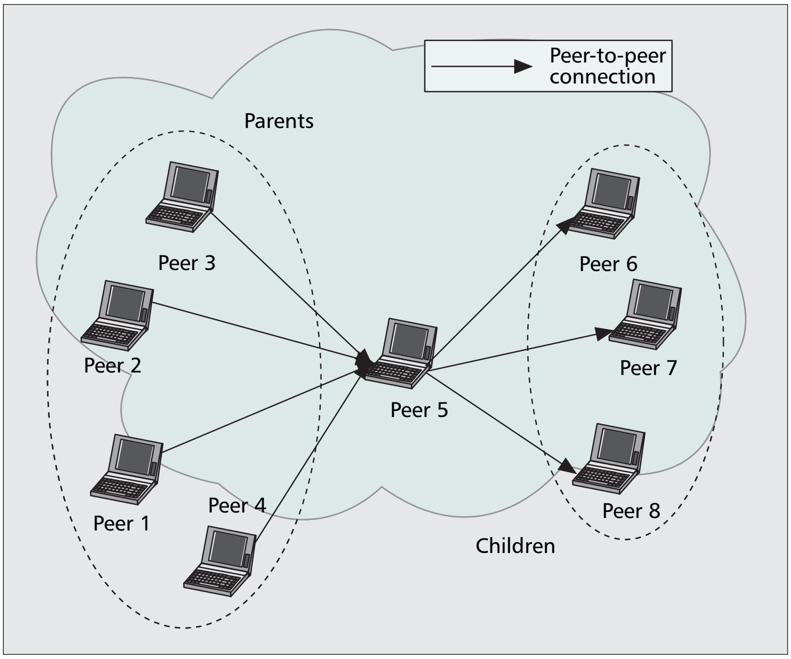
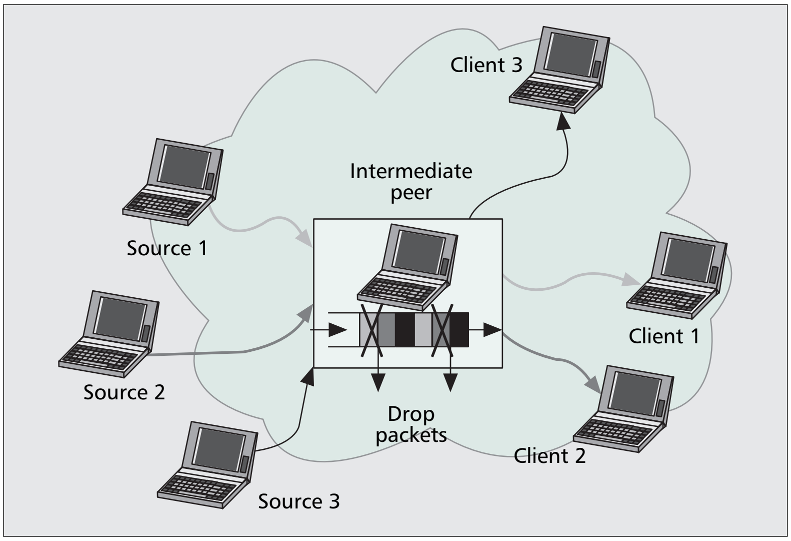


Figure 4. *Mesh architecture for media delivery in P2P systems.*

Distributed Path Computation

However, one of the major drawbacks of receiver-driven scenarios comes from the requirement for full topology knowledge at a single peer, namely the client. Only when the receiver knows the complete network topology (e.g., complete set of sources, along with their connection characteristics), can it make an optimal decision in terms of source peer selection and path rate allocation. However, as the network size increases, end-to-end traffic monitoring at a single peer becomes cumbersome and increasingly expensive or inefficient. Hence, moving at least part of the computation to intermediate peers becomes necessary. Augmenting the streaming scenario with intermediate peer functionality enables the maintenance of up-to-date information about network availability. The topology information is no longer relayed toward a single node in the scenario. Rather, every intermediate peer makes an individual routing decision for every incoming packet, based only on local topology information [10]. All incoming media flows arriving at one intermediate peer are relayed on the outgoing links, according to the specific forwarding rules implemented locally at the peer.

However, distributed path computation may result in sub-optimal streaming strategies, as no peer has complete knowledge of the network status. In heterogeneous network scenarios, the



■ **Figure 5.** *Intermediate peer processing of the incoming media streams. Based on distortion information present in the packet headers, a node can make a rate-distortion optimized scheduling decision for the incoming packets.*

results obtained by the locally-optimal decisions implemented at each intermediate peer may differ greatly from the global achievable optimal performance. Depending on the local path selection and the rate allocation rules implemented at each node, the media application can trade off the achieved average end-to-end quality with the flexibility and convergence time of the solution in case of network fluctuations. Finally, note that the routing of media packets in tree-based overlays is straightforward, as it is given directly by the structure of the multicast trees. However, the construction of these trees may use algorithms similar to routing solutions for multi-path networks. Even if the routing is simple in tree-based overlays, the proper scheduling of media packets is non-trivial, as exposed in the next section.

PACKET DELIVERY MECHANISMS IN P2P NETWORKS

RATE-DISTORTION EFFICIENT SCHEDULING

Packets of a media stream do not contribute evenly to the video quality at a receiving peer, and a packet is useful to the receiving peer only if:

- It arrives prior to its delivery deadline;
- All the previous packets required for its correct decoding were received already.

The unequal importance of video packets, along with timing constraints, naturally lead to the derivation of efficient packet scheduling algorithms that determine which packets should be forwarded at a given time instant, to maximize the overall streaming quality. The implementation of such scheduling strategies corresponds to a compromise between system complexity and rate-distortion efficiency.

Implementing completely distributed-video-packet scheduling algorithms in individual peers is a complex task. Ideally, these algorithms run independently on each source peer but unanimously decide the set of video packets to be sent, along with disjoint partitions allocated to each transmitting peer. Their goal is to maximize the quality of the received video stream, while avoiding the wasting of network resources. One way to reduce the real-time computational burden imposed by the distributed algorithms is to perform some initial off-line processing on the media stream. In the case of video-on-demand (VoD) applications, the benefit of each individual media packet can be computed and stored before the streaming session actually begins. Later, the scheduling mechanism performs a real-time selection of the set of packets to be transmitted, according to the available network resources and the relative importance of the packets. Furthermore, the complexity of joint scheduling in multiple source scenarios is alleviated by partitioning the set of media packets beforehand among the potential serving peers. Based on the feedback that each transmitter receives from the receiving client, it can perform independent scheduling decisions that are optimal in a rate-distortion sense while simultaneously ensuring that no packet is scheduled for transmission more than once across all participating sources [11]. Other solutions for solving packet scheduling issues in P2P networks are provided by in-network scheduling and queue management and packet coding for distributed delivery.

IN-NETWORK STREAM PROCESSING

Packet scheduling and queue management techniques can be enabled in video distribution trees, with the goal of distributively adapting the streaming process to the available network resources. A peer participating in a tree architecture is, in general, confronted with the situation where it must take a scheduling decision (whether to forward or rather drop packets) to maximize the video quality down-stream (Fig. 5). To guide this scheduling process, the rate-distortion information for each packet can be piggybacked in its header. Based on this side information, a forwarding node in the tree can optimally adapt or filter the passing video streams in a rate/distortion sense. In P2P systems, it is generally beneficial to distribute the video stream through multiple trees to deal with the typically unreliable P2P network structure, thereby taking advantage of the available network diversity. The scheduling process described previously extends easily for a single tree to multiple trees by sending disjoint subparts of the stream through separate trees [12]. This method, while very flexible and easy to implement at the intermediate forwarding peers, still requires some off-line processing of the media streams at the source.

CODING FOR DISTRIBUTED DELIVERY

As computing the relative benefit/importance of media packets is generally not trivial, coding methods were proposed as an alternative to scheduling algorithms. They consist mainly in smoothing out the differences in importance between media packets, thereby avoiding the need for complex packet scheduling. Channel codes can be employed to encode independent segments of a video stream, such as GOP [group

P2P system	Architecture	Media coding	Packet level scheduling
CoolStreaming	Overlay mesh	Possible use of layered coding or MDC	Yes
CoopNet	Multiple trees	MDC	No
PALS	Receiver-driven tree	Layered coding	Receiver-driven
PROMISE	Receiver-driven tree	Possible use of FEC	Receiver-driven
SPLIT Stream	Multiple trees	Possible use of MDC and FEC	No
Bullet	Overlay mesh	Possible use of MDC or FEC	Receiver-driven

Table 1. *P2P streaming systems' characteristics.*

The popularity of such systems and the continuous efforts of researchers in this area should bring efficient solutions to all these problems in the near future and enable a wide-scale deployment of high quality peer-to-peer streaming applications.

of pictures] and/or layers. For example, the authors in [13] propose to encode the substreams of a scalable video bitstream using Raptor codes. Raptor codes belong to the family of *rateless* or *Fountain* codes, which potentially allow an infinite number of coded symbols to be generated from a set of k source symbols. Any subset of $k + \varepsilon$ Raptor symbols (where ε is arbitrarily small) can then be used to decode the original k source symbols with high probability. Thus, by encoding an independently decodable subpart of k source symbols of the video stream with Raptor codes, the receiving client must retrieve merely enough of the corresponding Raptor symbols from all available serving peers on aggregate. The client can decode the subpart in question as soon as $k + \varepsilon$ symbols are received, without distinction of which particular packets are available to this end. This solution offers low decoding complexity and provides, along the way, a universal channel code for the transmitted stream. Finally, other applications of distributed coding solutions also can be proposed to improve the efficiency of media delivery from multiple source peers. Network coding, for example, is a recent technique where intermediate nodes forward linear combinations of incoming packets. This increases the capacity of the network (by reducing the amount of replication), as well as its resilience to packet loss [14], while coping with possibly different constraints at the end-user.

Discussion and Open Problems

We have examined the challenges of media streaming in P2P environments. Specific media adaptation techniques employed at the application layer and efficient network routing and packet scheduling mechanisms were considered, along with the most common P2P architectures, to address these difficulties. Some of these techniques are currently part of the most prominent P2P streaming systems implemented in the literature (see [1] and its references). Table 1 provides a brief overview of the main characteristics of these systems.

However, there are still several issues that must be resolved to establish P2P streaming as the premium media delivery solution. In particular, proper coordination of the aforementioned techniques across the different layers of the networking stack is not a trivial problem, and today many efforts are being deployed towards cross-layer optimization of multimedia communications. At the same time, large streaming scenarios featuring users with different (possibly, conflicting) constraints require specialized mechanisms to insure a stable system performance. Encouraging solutions are available in the area of network coding for P2P systems, differentiated services among peers [15], or node "coopetition" (a trade-off between peer cooperation and competition).

In addition, the development of efficient strategies to improve the stability of peer-to-peer topologies will offer an enhanced quality of service to users of streaming services, particularly in wireless environments. Solutions could be found in better mobility prediction models or in the development of self-healing systems, which are able to rapidly converge to stable topologies when peers join and leave the streaming session at random. Finally, digital rights management remains an important problem in peer-to-peer delivery, as it is still difficult to maintain secure media sessions in distributed and unstructured P2P topologies. However, the popularity of such systems and the continuous efforts of researchers in this area should bring efficient solutions to all these problems in the near future and enable a wide-scale deployment of high quality peer-to-peer streaming applications.

References

[1] X. Zhang *et al.*, "Coolstreaming/DONet: A Data-Driven Overlay Network for Efficient Live Media Streaming," *Proc. IEEE INFOCOM*, vol. 3, 13–17, Mar. 2005, pp. 2102–11.

[2] V. N. Padmanabhan, H. J. Wang, and P. A. Chou, "Resilient Peer-to-Peer Streaming," *Proc. IEEE ICNP*, Atlanta, GA, 2003.

[3] N. Magharei and R. Rejaie, "Understanding Mesh-Based Peer-to-Peer Streaming," *Proc. ACM NOSSDAV*, Newport, RI, 2006.

[4] Y. Shen *et al.*, "Peer-Driven Video Streaming: Multiple Descriptions Versus Layering," *Proc. IEEE ICME*, Amsterdam, The Netherlands, 2005.

[5] A. Albanese *et al.*, "Priority Encoding Transmission," *IEEE Trans. Info. Theory*, vol. 42, Nov. 1996, pp. 1737–44.

[6] L. Golubchik *et al.*, "Multi-Path Continuous Media Streaming: What Are the Benefits?" *ACM J. Perf. Evaluation*, vol. 49, no. 1–4, Sept 2002, pp. 429–49.

[7] J. Apostolopoulos *et al.*, "On Multiple Description Streaming with Content Delivery Networks," *Proc. IEEE INFOCOM*, vol. 3, 23–27, June 2002, pp. 1736–45.

[8] A. C. Begen *et al.*, "Multi-Path Selection for Multiple Description Video Streaming over Overlay Networks," *Sig. Processing: Image Commun.*, vol. 20, 2005, pp. 39–60.

[9] T. Nguyen and A. Zakhor, "Multiple Sender Distributed Video Streaming," *IEEE Trans. Multimedia*, vol. 6, no. 2, Apr. 2004, pp. 315–26.

[10] D. Jurca and P. Frossard, "Distributed Media Rate Allocation in Overlay Networks," *Proc. IEEE ICME*, Toronto, Canada, July 2006.

[11] J. Chakareski and P. Frossard, "Distributed Streaming Via Packet Partitioning," *Proc. IEEE ICME*, Toronto, Canada, July 2006.

[12] E. Setton, J. Noh, and B. Girod, "Rate-Distortion Optimized Video Peer-to-Peer Multicast Streaming," *Wksp. Advances in Peer-to-Peer Multimedia Streaming at ACM Multimedia*, pp. 39–48.

[13] J.-P. Wagner, J. Chakareski, and P. Frossard, "Streaming of Scalable Video from Multiple Servers Using Rateless Codes," *Proc. IEEE ICME*, Toronto, Canada, July 2006.

[14] C. Gkantsidis and P. R. Rodriguez, "Network Coding for Large Scale Content Distribution," *Proc. IEEE INFOCOM*, Miami, FL, Mar. 2005.

[15] C. Wu and B. Li, "Diverse: Application-Layer Service Differentiation in Peer-to-Peer Communications," *IEEE JSAC*, vol. 25, no. 1, Jan. 2007, pp. 222–34.

Biographies

DAN JURCA (dan.jurca@epfl.ch) received his B.Sc. degree from the Politehnica, University of Timisoara, Romania, in 2002. In 2003 he graduated from the graduate school at the Swiss Federal Institute of Technology (EPFL) in computer and communication sciences. He is currently working toward his Ph.D. degree inside the Signal Processing Institute of EPFL. He is a research assistant at EPFL, Lausanne. His research interests focus on adaptive rich media streaming (forward error correction, multipath media transmissions, and streaming in wireless environments).

JACOB CHAKARESKI received a B.S. degree from Ss. Cyril and Methodius University, Skopje, Macedonia, in 1996, an M.S. degree from Worcester Polytechnic Institute, Massachusetts, in 1999, and a Ph.D. degree from Rice University, Houston, Texas, in 2005, all three in electrical engineering. He performed his doctoral thesis research at Stanford University, Palo Alto, California, from 2002 to 2005. He is a post-doctoral researcher at EPFL. He received the best student paper award at VCIP 2004. He has held industrial research positions with Microsoft and Hewlett-Packard. He has co-authored over 50 international publications and has two pending patent applications. His research interests include networked media systems, statistical signal processing, and communication theory.

JEAN-PAUL WAGNER received his M.Sc. in communication systems from EPFL in 2004. In May 2004 he joined the Signal Processing Institute as a research assistant, where he is currently working toward a Ph.D. degree. His research interests focus on distributed media stream over best-effort networks.

PASCAL FROSSARD (pascal.frossard@epfl.ch) received his M.Sc. and Ph.D. from EPFL in 1997 and 2000, respectively. He has been an assistant professor at the Signal Processing Institute of EPFL since 2003. Between 2001 and 2003 he was with the IBM T. J. Watson Research Center, Yorktown Heights, New York. His current research interests include image representation and coding, nonlinear representations, visual information analysis, joint source and channel coding, multimedia communications, and multimedia content.