Adaptive Systems for Improved Media Streaming Experience

Jacob Chakareski and Pascal Frossard, Ecole Polytechnique Fédérale de Lausanne

ABSTRACT

Supporting streaming media applications over current packet network infrastructures represents a challenging task in many regards. For one, the lack of quality of service (QoS) guarantees in existing networks such as the Internet means that time-constrained media packets will face dynamic variations in bandwidth, loss rate, and delay as they traverse the network from the sender to the receiver. The variable rate of media traffic represents yet another difficulty when transmission constraints need to be met. Finally, the heterogeneity of client devices and access bandwidth coupled with custom user preferences exacerbate the problem of smooth and quality-optimized media playback even further. In this article we provide an overview of the various techniques for media and streaming strategy adaptation, which can be employed to deal with the difficulties imposed by such dynamic environments. These techniques depend on the characteristics of the media application, in particular on the network streaming infrastructure and the timing constraints imposed on the media packets' delivery. We survey adaptation techniques that act on the encoding of the multimedia information, on the scheduling of the media packets, or that try to combat transmission errors. We also briefly overview some media-friendly networking solutions, which contribute to increased QoS by incorporating some level of intelligence in intermediate network nodes. Finally, we describe a few open challenges in media streaming, emphasizing strategies based on promising cross-layer approaches where adaptation strategies are applied in a coordinated manner, across different layers of the network protocol stack.

INTRODUCTION

Since the early developments of the Internet a couple of decades ago, heterogeneity among components of transmission systems has been continuously growing. Access bandwidth offered to clients can be highly varying, and the type of access can be as different as wired broadband ADSL, or wireless 3GPP services, for example. And for the same multimedia service, customer devices may also have quite different capabilities due to the rapid technology evolution in the industry of computing platforms.

In addition, packet networks employed for transmitting media streams today are predominantly best-effort in nature. In other words, there are no guarantees on deliveries of packets sent across networks like the Internet, as these infrastructures typically exhibit variable bandwidth, packet loss, and delays. In addition, such fluctuations are accentuated by the increasing mobility of users and their changing preferences or customization in the multimedia application. Lastly, media packet streams are not homogeneous either, since information sources are typically nonstationary and encoding schemes generate media packets that often obey a hierarchical structure with decoding dependencies between packets.

All these parameters render the problem of streaming media quite challenging due to the inherent timing constraints associated with the playback of data units in a media presentation. One has to optimize the quality experienced by a user, while simultaneously addressing the varying transmission conditions. Fortunately, media streams exhibit a certain degree of tolerance to data loss, as well as to controlled variations of their bit rate. These properties are exploited for designing streaming systems that adapt to the network conditions in order to offer an adequate quality to the media client. Adaptivity is the only way to design efficient systems in conditions where static design choices would dramatically fail to provide an acceptable quality to the end user. Appropriate coding choices, smart packet selection, and efficient scheduling are the key components of effective adaptive solutions.

In essence, all the adaptivity techniques proposed thus far attempt to make the streaming process responsive and therefore more reliable by appropriately modifying the behavior of the different constituent components of a media streaming system. They can be applied at the server (sender), at the client (receiver), at intermediate network nodes (proxy), or in a coordinated manner between these agents. The choice of an adaptive streaming solution is mostly driven by the specific characteristics of the multimedia application. We group these solutions in adaptive media processing techniques, adaptive

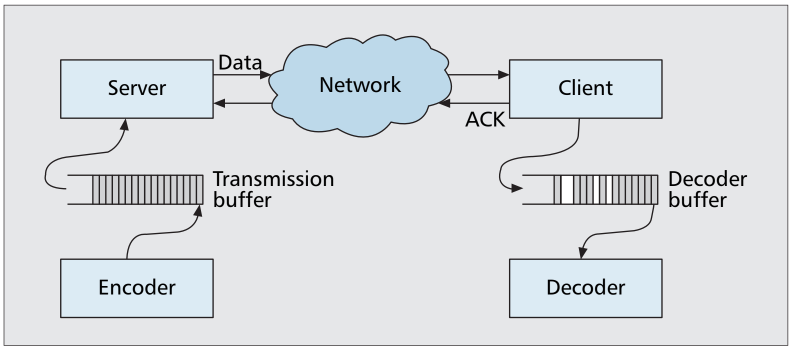
0163-6804/07/$20.00 © 2007 IEEE

streaming strategies, and media-friendly networking solutions. Finally, we describe a few open challenges for streaming time constrained data units in adverse environments, with a particular emphasis on the promising cross-layer schemes, where effective cooperation between layers brings further benefits in terms of performance.

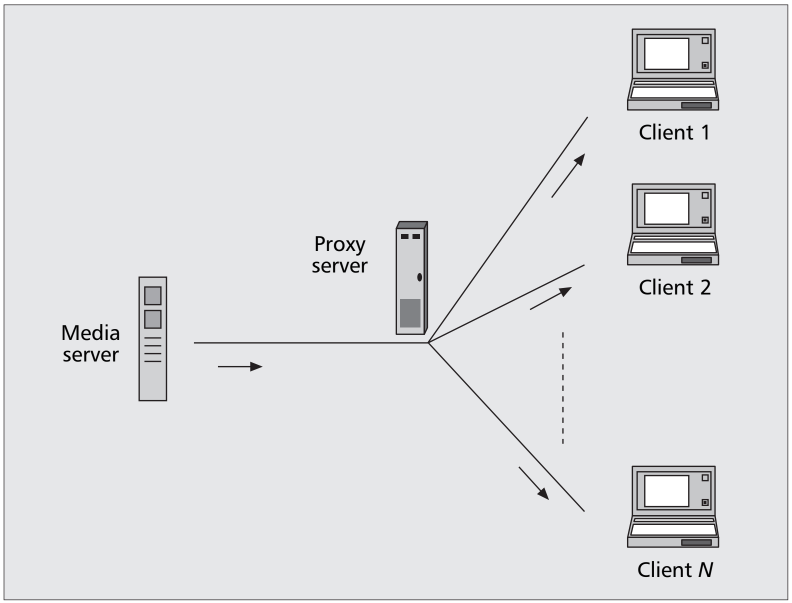
COMMON MEDIA STREAMING ARCHITECTURES

This section briefly introduces the common streaming architectures that are deployed today. The particular type of architecture determines in general the set of possible solutions that can be designed for adaptive streaming. Here, we only consider scenarios that involve a single multimedia asset; the case of multiple assets can be considered basically as a superposition of several single asset scenarios, where the common resources have to be judiciously shared between the different media applications.

The one-to-one model (single sender–single receiver) is certainly the most common streaming scenario encountered in practice. It comprises a sending server that transmits data units of a media presentation over a packet network to a receiving client application (Fig. 1). Media data, encoded live or pre-encoded off-line, are packetized into data units, and put in a sending buffer, or written to a special streaming media file, respectively. When requested by the client, the server reads the data units from the file and transmits them over the network. The client, upon receipt of the data units, buffers them in a decoder buffer. At the appropriate time, the client sends data units from the buffer to the media decoder, for decoding and presentation. The client generally starts to decode the media stream after a predefined delay, called the playback delay. The purpose of this delay and the decoder buffer is two-fold: to remove network delay jitter and to allow for recovery of lost packets using positive acknowledgments (ACKs), when possible. In one-to-one scenarios, the coding and streaming mechanisms can be finely tuned to provide an optimized quality of service to the receiver, and precise adaptation to both the actual state of the network and the receiver can be provided.



■ **Figure 1.** *Block diagram of a typical streaming system.*



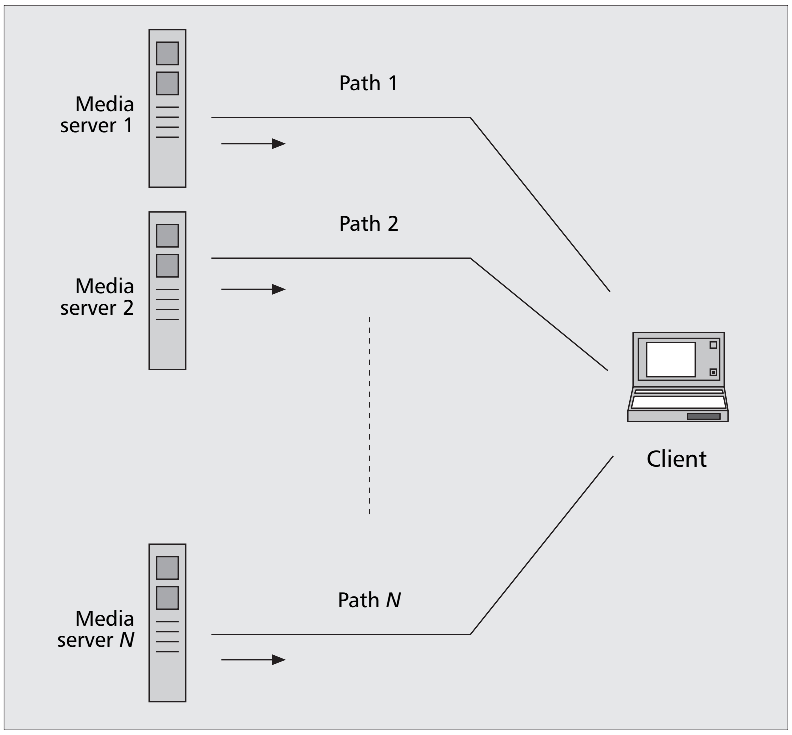
■ **Figure 2.** *Streaming system with a single sender and multiple receivers.*

The second streaming scenario is a typical broadcast architecture with a single sender and multiple receivers, as illustrated in Fig. 2. In this setup, a single sender streams media data to multiple receiving clients, possibly over a common network path. The most important issue related to this setting is how the available bandwidth on the shared path should be distributed between the respective information sent to the different clients. Adaptive solutions generally rely on different versions of a media stream, or on scalable bitstreams. Fine adaptation to the client is generally not possible in one-to-many scenarios, and compromises have to be derived to offer a good overall performance to the client population.

A third streaming scenario based on multiple senders and a single receiver (Fig. 3) is becoming increasingly popular due to the deployment of distributed and peer-to-peer applications. Here, media packets are sent from different servers over separate network paths to a single client running at the receiver. Whether and how the transmission schedules of the sending servers should be coordinated is the key issue specific to this setting. The two extreme cases are either no coordination at all, where each of the servers makes transmission decisions regardless of the actions of the other servers, or a complete synchronization, where the scheduling actions of the sending servers are performed in concert.

Generally, streaming media involves transporting multimedia information through a number of intermediate nodes and edge servers, before reaching the end-users. These intermediate nodes can be used to support the delivery of continuous media streams. Efficient distribution schemes take benefit of the processing capabilities of intermediate nodes in order to finely adapt the media streams to the varying network conditions and client access technologies. This can be achieved by application-level multicast, content replication, and network transcoding or filtering. These latter operations may in general be expensive if the encoding process has not been designed accordingly. Therefore, coordination between the different parts of a streaming system is important, for example, progressive or scalable coding of the media information would facilitate the filtering (rate adaptation) process within the network nodes. Finally, content delivery networks (CDNs) can be employed to provide many of these functionalities to media streaming users. These networks comprise collections of edge servers strategically placed at different locations in the Internet such that a client can chose the server that results in shortest round-trip time and least amount of congestion.

Adaptive streaming algorithms can furthermore be classified into receiver-driven, proxy-driven, and sender-driven solutions, depending on the party that controls the behavior of the system. In addition, streaming applications can be classified into two main categories: live and on-demand services. This coarse distinction is mainly dependent on the timing constraints and the degree of interactivity offered to the users. The specific characteristics of the multimedia streaming application drive the choice of the tools that can be used to optimize the quality of service in dynamic network conditions.



■ **Figure 3.** *Streaming system with multiple senders and a single receiver.*

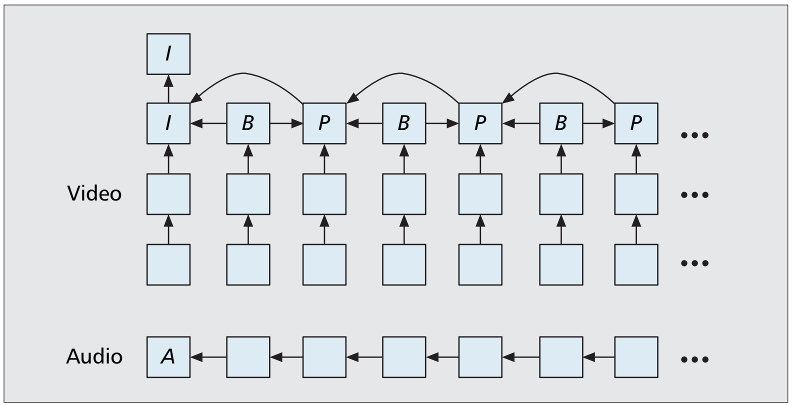
Network-Adaptive Media Processing

In order to enable streaming solutions that can adapt to the network state and/or to the receiver capabilities, systems often rely on network-adaptive media coding algorithms, or adaptive decoding strategies. Early media coding algorithms have been designed exclusively for achieving a high compression ratio, without taking into account any streaming related aspects. However, it rapidly became obvious that efficient streaming systems have to rely on coding methods that provide some support for packet transmission over imperfect channels. Therefore, the latest coding schemes explicitly consider transmission related effects on the media information when building the compressed bitstream. These algorithms encode and packetize the media information under a form that facilitates adaptation to the network characteristics, expressed in terms of bandwidth variation or packet loss. Such techniques include for example scalable encoding, efficient bitstream packetization, error-resilient encoding, and dynamic changes of compressed data units' dependencies. On the decoder side, concealment techniques or adaptive play-out algorithms additionally contribute to mask lost or delayed information.

Rate Adaptive Media Coding

Scenarios where an encoder can compress a stream with a rate profile that exactly fits the channel bandwidth variations are quite rare in practice, mostly because the channel bandwidth is not perfectly known at the time of encoding. Therefore, having a scalable representation of the media stream becomes a very attractive feature as it offers flexibility for rate adaptation. Scalability issues include adaptation to the receiver in terms of spatio-temporal resolution and computing power, to the available bandwidth, and to the multimedia content according to user preferences. Scalable encoding provides an elegant way of achieving bandwidth adaptation by rate scalability of the media stream, which is commonly organized into hierarchical quality layers, as illustrated in Fig. 4. In particular, scalable coding is beneficial for on-demand media applications, where the content is encoded only once, and then is eventually served at multiple resolutions, depending on the client access bandwidth and the network state at the time of transmission. Furthermore, scalability is particularly attractive in many applications based on the one-to-many architectures where the same media stream, or a finite set of streams, has to serve different clients, each with its own characteristics. In order to limit the computational load on the streaming server in such scenarios, intrinsically scalable streams facilitate filtering or rate adaptation operations in intermediate network nodes. Layered media representations have become part of the latest joint standardization efforts of the ITU and the ISO, where the first draft of the international standard on scalable video coding (SVC) is expected to be released this year. However, standard layered encoding still provides in general smaller compression efficiency relative to nonscalable encoding.

Limited rate scalability is also achieved by switching (during streaming) between multiple independent encodings of the same media content, in response to varying network bandwidth, as implemented for example in the SureStream system of Real Inc. [1]. The switching can be done at independently encoded data units (the so-called intra- or I-frames in MPEG terminology) or at specially designed anchor frames, e.g., SP-frames as provided by a provision of the latest video coding standard, H.264. Finally,



■ **Figure 4.** *Scalable audio-video presentation.*

transcoding is another alternative solution for rate scalability, where a nonscalable media stream is simply reencoded in order to adapt its bit-rate to the available resources. However, this approach is quite greedy in terms of computational complexity, which makes its application quite limited in practice.

Loss Adaptive Media Coding

Network-adaptive media coding also provides a solution to packet loss that may arise during a streaming session. Recent coding algorithms include several error resiliency tools that limit the impact of transmission losses and the resulting error propagation effects, which are inherent to the predictive coding nature of most media compression schemes. Flexible macroblock ordering, multireference frame motion estimation, and redundant slices, are some of these tools proposed in the recent H.264 standard.

One of the causes of error propagation within media streams is the mismatch between media data units and network transport entities. Efficient packetization algorithms can help to limit the effect of losses by providing means to build packets that are decodable independently of the reception of other packets. In order to further combat error propagation due to temporal prediction, error-resilient encoding includes dynamic switching of the encoding mode (Intra or Inter) for macroblocks of video frames that trades off compression efficiency for error resiliency [2]. An equivalent technique that deals with error propagation, dynamically controls the prediction dependencies between video frames (e.g., reference picture selection, RPS). A related approach is video redundancy coding (VRC) from the H.263+ video coding standard, where a video sequence is first subsampled into *K* complementary subsequences (or threads) of video frames in a round-robin fashion, which are then encoded independently.

In addition, media information segments can also be encoded differently, depending on their relative importance. Due to the nonstationary nature of the multimedia content and to the hierarchical structure of most common coders, all the media bits are not equal: some parts of the bitstream are more important than others in terms of contribution to the overall quality. The source and channel coding separation theorem of Shannon is only justifiable for arbitrarily long code block lengths and arbitrarily high coding complexity. These requirements, however, are not met in multimedia communication with timing constraints. The source and channel encoders have to be built jointly, taking into account the a priori unbalanced importance of multimedia data. Thus, an optimal error protection scheme has to understand the bitstream content, and to preferably protect the most important bits in order to ensure graceful quality degradation as the packet loss rate increases. Joint source and channel coding schemes have been introduced to take benefit of the knowledge of the relative data importance and to optimally balance the error protection within the stream [3]. The most complete joint source and channel coding schemes even take into account hypothetical loss effects, and mimic the behavior of the decoder (e.g., error concealment) to globally optimize the end-to-end quality [4].

However, finely adaptive error protection is difficult to design in scenarios where the loss behavior is hard to predict, or where the access bandwidth is quite heterogeneous among clients. This problem can be alleviated by creating versions of the media bitstream with different degrees of resiliency to loss, for different predefined networking conditions. Another alternative is offered by Multiple Description Coding, which is less sensitive to channel prediction mismatch than classical joint source and channel coding methods. It basically consists in leaving a controlled degree of redundancy in the media stream, so that decoders obtain a quality of service directly driven by the number of received packets. The best signal reconstruction is obtained when all descriptions are correctly received, while the correct reception of a single description should already provide a reasonable quality [5]. The main problem lies in the construction of the signal descriptions which should neither be completely independent to ensure error-free delivery of the main information content, nor completely redundant to constrain the bit rate. In addition, ensuring synchronicity between the encoder and decoder states in video applications with motion estimation is not a trivial issue.

Passive Receiver-Based Techniques

The audiovisual experience of a media presentation can be further improved by performing passive techniques at the receiver when the compressed media data is decoded in order to be (dis)played by the client application. In particular, error concealment [6] is employed to replace missing data units at decoding. This allows for continuous decoding of the compressed media stream and improves the quality of the media presentation. A simple form of concealment is to replace a missing data unit with its temporally nearest predecessor data unit that has been received and decoded on time. Alternative and more sophisticated approaches include, for example, bidirectional interpolation of a data unit from its predecessor and successor data units. Finally, adaptive media play-out adjusts the playback speed of the media presentation at the receiving client in order to prevent buffer underflow and overflow events [7].

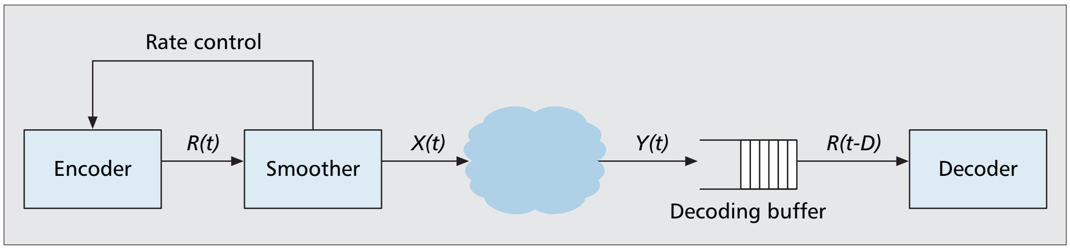


Figure 5. *Adaptive packet scheduling: As the output of the network* Y(t) *is generally unknown, application-layer QoS control can adapt the sending trace* X(t)*, possibly jointly with the source trace* R(t)*, to get the best quality at the receiver.*

Judicious error control strategies can thus be enabled due to the knowledge of the media stream properties at the application layer. They mainly include include retransmission (ARQ), forward error correction (FEC), or hybrid combination of both.

Adaptive Streaming Strategies

Application-layer QoS control techniques are used to deal with dynamically varying network conditions that can lead to significant data rate variations or unexpected packet losses. These solutions include in major part adaptive error control and rate control schemes. Since the media units are unequally important, error protection is chosen such that delivery of the most important information segments is ensured. Similarly, another important group of adaptive streaming solutions relies on judicious packet selection and scheduling when resources become scarce. Proper control of the source and sending rates allows to respect bandwidth constraints, and hence to avoid congestion onsets.

Adaptive Error Control

Error control aims to recover lost packets or at least to limit the effects of degradation on the perceived quality. In the early beginnings, error control used to be performed by the transport layer in the Internet. Historically, communication links have been designed for error-free and possibly delayed data transmission. They do not take into account the properties specific to visual communications, such as the strict delay requirements and the certain tolerance to channel loss due to the redundancy left in the compressed signal. Therefore, error control has been shifted to the upper layers of the end-to-end system, while timing control is left to the transport or RTP protocols. Judicious error control strategies can thus be enabled due to the knowledge of the media stream properties at the application layer. They mainly include include retransmission (ARQ), forward error correction (FEC), or hybrid combination of both.

One of the most common application layer techniques in practical streaming systems is Automatic Repeat reQuest (ARQ), where lost packets are recovered through retransmissions [8]. ARQ systems use combinations of time-outs and positive and negative acknowledgments to determine which packets should be retransmitted. They can allow for prioritized retransmission in the case of layered video. ARQ has the advantage of guaranteeing error-free delivery, but at the price of a possibly large and unpredictable delay. Such schemes may not be appropriate for applications with very tight delay constraints, or in broadcast scenarios due to the bandwidth explosion phenomenon that arises when the states of the receivers are not synchronized.

FEC means that redundancy is added to the data so that the receiver can recover from losses or errors without any further intervention from the sender. Considering the delay requirements for interactive or real-time media streaming applications, FEC is generally more appropriate than retransmission, even if it does not guarantee an error-free packet delivery. As long as a sufficient number of packets is received, missing media packets can be recovered at the receiver with the help of redundancy packets, which are generally built from block codes (e.g., Reed–Solomon codes). The redundancy packets can be organized to provide unequal error protection (UEP) for different layers of a (scalable) media representation [9] in order to achieve an improved average quality in error-prone environments.

Adaptive Packet Scheduling

Proper packet scheduling is also important for the performance of an adaptive system: it allows for choosing the most important data units to be transmitted, given actual timing constraints, and for controlling the streaming rate in order to avoid potential congestion in the network (Fig. 5). Optimized packet scheduling at the application layer takes into account data units' dependencies and importance for the reconstruction of the media stream at the receiver when performing transmission decisions for media packets [10]. The media bitstream is abstracted as a directed acyclic dependency graph in which each node represents a packetized data unit, and each packet is labeled with its timestamp, size, and importance. In this setting, a set of transmission choices is considered, where each choice is associated with a cost per byte of transmitting a media packet, and an error probability of not delivering the packet before its deadline. A rate-distortion optimization formulation can then be proposed to define the best (re)transmission strategy, which maximizes the quality at the decoder.

Congestion control further helps in preventing packet loss and reducing delays by carefully limiting the bandwidth available to the sender. Media streams often exhibit highly varying bit rates due to the nonstationary nature of the multimedia content, which makes the management of transmission bandwidth involved. Several rate-control or scheduling techniques can be applied to a media stream to address this issue. For example, the sender can take advantage of the playback delay to smooth or shape the stream and make it compliant with the available

Media streams often exhibit highly varying bit rates due to the non-stationary nature of the multimedia content, which makes the management of transmission bandwidth involved. Several rate-control or scheduling techniques can be applied to a media stream to address this issue.

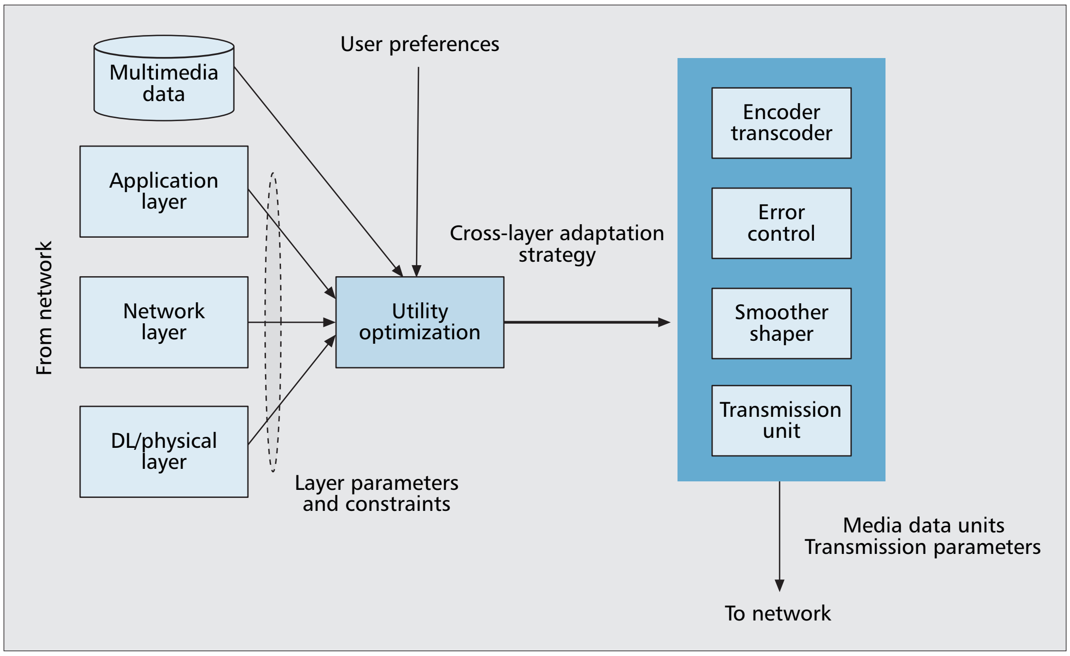


Figure 6. *Cross-layer media adaptation techniques.*

bandwidth and the client device capabilities. In addition to reducing the variability of the sending rate, proper scheduling also aims at minimizing the playback delay and decoding buffer levels [11]. A related effort, TCP-friendly rate control (TFRC), also provides a lower variation of throughput over time relative to TCP, while simultaneously allowing for fair sharing of the available bandwidth with competing TCP flows. This makes TFRC more suitable for streaming media compared to TCP. Still better performance can be achieved when the scheduling is performed jointly with appropriate packet selection or source rate control mechanisms. The joint source and sending rate selection allows to achieve optimized performance, while respecting constrained bandwidth resources [12, 13].

Media-Friendly Networking

Efficient adaptive streaming solutions may also benefit from any form of network or link layer support, which can offer more than a simple blind packet transport. The network infrastructure should be able to judiciously route media data, and transcode or filter media streams. Media-friendly networking techniques include for example selecting (potentially dynamically) an appropriate network path/route for streaming the media packets from the server to the client, and providing multiple network routes through which media packets can be sent (i.e., the so-called path diversity [5]), in the case of a single sender, and the so-called server diversity, in the case of multiple sending servers. Frequently, path diversity has been studied for streaming over multihop mobile wireless networks in order to provide higher bandwidth and robustness to end-to-end connections [14]. Receiver-driven streaming of media packets from multiple servers to a single client has been studied for the first time in [15], where rate control and packet partition algorithms have been proposed to coordinate the transmissions from the multiple servers.

While the previous sections have mostly considered packet losses due to congestion, wireless streaming scenarios also raise the issue of transmission errors, since the physical medium is characteristically unreliable. Therefore, in order to combat these errors techniques such as error-correction coding, link layer retransmission (repetition coding), adaptive modulation, and power control have been introduced. In particular, error correction coding introduces a certain amount of redundancy in the data bits sent over the physical medium so that they can be recovered at the receiver in case some of them get corrupted during transmission. If a received packet contains a larger number of corrupted bits than the error-correction code allows to be recovered, the packet is discarded at the receiver as unrecoverable. Then, retransmission of this packet at the link layer is employed to ensure its delivery to the receiver. Power control, on the other hand, ensures that transmission errors do not occur in the first place, by employing higher transmission power for sending packets when the communication channel conditions are particularly adverse. All these three techniques can be applied adaptively, that is, the amount of redundancy, the number of retransmissions, and the amount of power can be dynamically adjusted based on the present channel conditions and user preferences, for example, as in the latest standard for mobile wireless devices (3GPP).

Challenges

Despite the promising advances achieved by adaptive streaming in optimizing the quality of service of networked media applications, there are still many open challenges that need to be addressed. For example, the development of large-scale, multiuser systems, necessitates high scalability and optimal resource allocation between users and applications. As the utility per bit may dynamically vary across different applications, distributed bandwidth allocation in large networks is still an unsolved problem. Therefore, a lot of the research efforts at present focus on providing efficient scalable media representations, as well as effective solutions for a fair distribution of resources between concurrent media streaming applications. Additionally, adaptive streaming systems generally assume to have a good knowledge of the network status. However, this assumption is quite strong in practice, and the design of solutions with reduced sensitivity to precise knowledge of the actual network resources is still under investigation.

One of the most promising strategies for adaptive streaming certainly lies in the development of genuine cross-layer algorithms, which optimally combine the mechanisms enabled at different layers of the transmission system. These techniques arise in situations where adaptation of the streaming process is performed jointly over more than one layer of the network hierarchy stack, as illustrated in Fig. 6. Specifically, a cross-layer framework consists of jointly analyzing, selecting, and adapting the different techniques available at the individual layers. Application of cross-layer approaches has been particularly noticeable in the case of wireless networks, which provide a wider range over which such joint adaptations can be performed. By performing optimization across multiple layers, enhanced performance is obtained relative to the case when optimizations are performed separately over the individual layers. However, the complexity of the global optimization problem is overly constraining in general and, therefore, most of the work in crosslayer design considers suboptimal yet very efficient solutions, with a limited set of optimization parameters. Examples of cross layer schemes include, for example, joint strategies for rate allocation at the network layer and packet scheduling at the application layer, congestion and admission control at the network layer performed in concert with traffic prioritization at the application layer, and so forth. An excellent review of cross-layer principles for multimedia transmission is provided in [16].

The complexity of the global optimization problem is overly constraining in general, and therefore most of the work in crosslayer design considers suboptimal yet very efficient solutions, with a limited set of optimization parameters.

References

[1] G. J. Conklin *et al.*, "Video Coding for Streaming Media Delivery on the Internet," *IEEE Trans. Circuits and Sys. Video Tech.*, Special Issue on Streaming Video, vol. 11, no. 3, Mar. 2001, pp. 269–81.

[2] R. Zhang, S. L. Regunathan, and K. Rose, "Video Coding with Optimal Inter/Intra-Mode Switching for Packet Loss Resilience," *IEEE JSAC*, vol. 18, no. 6, June 2000, pp. 966–76.

[3] K. Stuhlmüller, N. Färber, M. Link, and B. Girod, "Analysis of Video Transmission over Lossy Channels," *IEEE JSAC*, vol. 18, no. 6, June 2000, pp. 1012–32.

[4] G. Coté, S. Shirani, and F. Kossentini, "Optimal Mode Selection and Synchronization for Robust Video Communications over Error-Prone Networks," *IEEE JSAC*, vol. 18, no. 6, June 2000, pp. 952–65.

[5] J. Apostolopoulos and M. Trott, "Path Diversity for Enhanced Media Streaming," *IEEE Commun. Mag.*, vol. 42, no. 8, Aug. 2004, pp. 80–87.

[6] Y.Wang and Q. Zhu, "Error Control and Concealment for Video Communications: A Review," *Proc. IEEE*, vol. 86, no. 5, May 1998, special issue on Multimedia Signal Processing, pp. 974–97.

[7] R. Ramjee *et al.*, "Adaptive Playout Mechanisms for Packetized Audio Applications in Wide-Area Networks," *Proc. IEEE INFOCOM*, vol. 2, Toronto, Ontario, Canada, June 1994, pp. 680–88.

[8] H. Liu and M. E. Zarki, "Performance of H.263 Video Transmission over Wireless Channels using Hybrid ARQ," *IEEE JSAC*, vol. 15, no. 9, Dec. 1999, pp. 1775–86.

[9] R. Hamzaoui, V. Stanković, and Z. Xiong, "Optimized Error Protection of Scalable Image Bit Streams: Advances in Joint Source-Channel Coding for Images," *IEEE Signal Proc.*, vol. 22, no. 6, Nov. 2005, pp. 91–107.

[10] P. A. Chou and Z. Miao, "Rate-Distortion Optimized Streaming of Packetized Media," *IEEE Trans. Multimedia*, vol. 8, no. 2, Apr. 2006, pp. 390–404.

[11] J. Rexford and D. Towsley, "Smoothing Variable-Bit-Rate Video in an Internetwork," *IEEE/ACM Trans. Net.*, vol. 7, no. 2, Apr. 1999, pp. 202–15.

[12] C.-Y. Hsu, A. Ortega, and A. Reibman, "Joint Selection of Source and Channel Rate for VBR Video Transmission under ATM Policing Constraints," *IEEE JSAC*, vol. 15, no. 5, Aug. 1997, pp. 1016–28.

[13] O. Verscheure, P. Frossard, and J. L. Boudec, "Joint Smoothing and Source Rate Selection for Guaranteed Service Networks," *Proc. INFOCOM*, vol. 2, Anchorage, AK, Apr. 2001, pp. 613–20.

[14] S. Mao *et al.*, "Video Transport over Ad Hoc Networks: Multistream Coding with Multipath Transport," *IEEE JSAC*, vol. 21, no. 3, Dec. 2003, pp. 1721–37.

[15] T. Nguyen and A. Zakhor, "Distributed Video Streaming over Internet," *Proc. Multimedia Comp. and Net.*, vol. 4673. San Jose, CA: SPIE, Jan. 2002, pp. 186–95.

[16] M. van der Schaar and S. Shankar, "Cross-Layer Wireless Multimedia Transmission: Challenges, Principles, and New Paradigms," *IEEE Wireless Commun.*, vol. 12, no. 4, Aug. 2005, pp. 50–58.

Biographies

JACOB CHAKARESKI (jakov.cakareski@epfl.ch) is a postdoctoral researcher at the Swiss Federal Institute of Technology (EPFL), Lausanne. He 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 and computer engineering. He performed his doctoral thesis research at Stanford University, Palo Alto, California, from 2002 to 2005. He received the best student paper award at the VCIP 2004 conference. He has held research positions with Microsoft and Hewlett-Packard. He has coauthored over 50 international publications and has two pending patent applications. His research interests include networked media systems, statistical signal processing, communication theory, and Macedonian history.

PASCAL FROSSARD (pascal.frossard@epfl.ch) has been an assistant professor at the Signal Processing Institute of EPFL since 2003. He got his M.Sc. and Ph.D. degrees, both from EPFL, in 1997 and 2000, respectively. 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 distribution. He was General Chair of IEEE ICME 2002, Lausanne, Switzerland, and has been a member of the organizing or technical program committees of numerous conferences. He has served as Guest Editor of special issues on Streaming Media (*IEEE Transactions on Multimedia*), Media and Communication Applications on General Purpose Processors: Hardware and Software Issues (*Journal of VLSI SPSS*), and Image and Video Coding Beyond Standards (Elsevier *Journal of Signal Processing*). He is an Associate Editor of *IEEE Transactions on Multimedia* (2004–), and served as a member of the Editorial Board of *EURASIP Journal of Applied Signal Processing* (2003–2005). Since 2004 he has served as Vice-Chair of the IEEE Multimedia Communications Technical Committee and as a member of the IEEE Multimedia Signal Processing Technical Committee. He received a Swiss NSF professorship award in 2003 and the IBM Faculty Award in 2005.