1

Introduction

Since the appearance of Napster in early 1999, peer-to-peer (P2P) networks have experienced tremendous growth. In 2003, P2P became the most popular Web application, and, at the end of 2004, P2P protocols represented over 60% of the total Internet traffic, dwarfing Web browsing, the second most popular application [1]. This rapid success was fueled by file transfer networks which allow users to swap media files, despite the large latency often necessary to complete a download. It is expected to continue at a fast pace, as new compelling P2P applications are developed. One of these applications, P2P multicast, is explored in this book.

In P2P multicast, a media stream is sent to a large audience by taking advantage of the uplink capability of the viewers to forward data. Similar to file transfer networks, data propagation is accomplished, via a distributed protocol, which lets peers self-organize into distribution trees or meshes. The striking difference is that this should happen in real-time, to provide all connected users with a TV-like viewing experience. Compared to content delivery networks, this approach 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.

To become widely adopted, P2P streaming systems should achieve high and constant video quality, as well as low startup latency. Three factors make this a difficult task. First, the access bandwidth of the peers is often insufficient to support high quality video. Second, the peers may choose to disconnect at any time breaking data distribution paths. This creates a highly unreliable and dynamic network fabric. Third, unlike in client-server systems, packets often need to be relayed along long multi-hop paths, each hop introducing additional delay, especially when links are congested. This unique set of challenges explain why early implementations, although they constitute remarkable progress and demonstrate the feasibility of large scale P2P streaming, fall short of the goals.

In our own research, which is presented in this book, we have investigated video streaming systems in order to enhance the perceived image quality,

increase their robustness to errors, and reduce latency. As the techniques that we are considering have merit for both client-server and P2P networks, our approach is to analyze the performance of unicast systems first, before extending and adapting the algorithms to P2P multicast. We investigate, in particular, throughput-limited environments where video streams may cause self-congestion when their rate is too high or when their transmission is inadequately controlled. In this context, we show the importance of adaptive packet scheduling which may help extend the range of sustainable rates, reduce startup latency, and maintain high error resilience.

The rest of this book is organized as follows. In the next chapter, we describe recent advances in the field of video compression, video streaming, multicast architectures and P2P systems, related to our work. In Chapter 3, we consider client-server systems. We focus on the impact of self-inflicted congestion on low-latency video streaming. We present an end-to-end rate-distortion model which captures the impact of both compression and late loss due to self-inflicted congestion. The model is helpful for deriving an encoding rate which maximizes video quality. In the second part of the chapter, we introduce the concept of congestion-distortion optimized (CoDiO) packet scheduling. Different from rate-distortion optimization, this type of algorithm determines which packets to send, and when, to maximize decoded video quality while limiting network congestion. We describe the operations of this scheduler, and of a low-complexity scheduler derived from it, and analyze their performance over a simulated network. The experimental results presented in this chapter are the first in-depth comparison of congestion and rate-distortion optimized schedulers.

In Chapter 4 and in Chapter 5, we consider the scenario of live P2P multicast where the video stream is sent to a large population of peers. Before showing how adaptive streaming algorithms can be adapted to this context, we describe a distributed control protocol, designed for fast startup, which is run by the peers to construct and maintain multiple multicast trees rooted at the video source. This allows to transmit a video stream synchronously to a set of peers by relying on their forwarding capacity. The operations of this protocol are described in detail and an analysis of its performance over different networks is presented. In Chapter 5, we explain how to extend congestion-distortion optimized packet scheduling to P2P live streaming to further reduce startup latency and to support higher rates. Similar to the CoDiO scheduler, the adaptive scheduler we present transmits in priority packets which contribute most to the decoded video quality. In addition, it favors peers which serve, subsequently, larger populations since they have the largest impact on the overall video quality. This one-to-many packet scheduler is combined with a retransmission scheduler which operates at the receivers to request, in priority, missing packets which will lead to the largest distortion reduction. We investigate the performance of this streaming technique over simulated networks of hundreds of peers. Conclusions and future research directions are

presented in Chapter 6. The appendix contains additional technical details about the different video streaming experiments reported previously in the book.

2

Background

The purpose of the work presented in this book is to analyze and improve the performance of video streaming systems operating in bandwidth-constrained networks. In particular, we consider low-latency applications where a source is serving a single receiver or where video is multicast to a population of peers. Our work builds upon recent advances which have focused on providing better compression efficiency, on increasing the robustness of video streaming systems, and on building efficient multicast architectures or peer-to-peer systems. In the following, we present an overview of the state-of-the-art in these areas.

2.1 Video Compression

2.1.1 H.264 Video Coding

The results we present in the following chapters were obtained for video sequences compressed using the latest video coding standard H.264, also called MPEG-4 Advanced Video Coding or H.264/AVC, which was finalized in March 2003 [2]. Like its predecessors, H.261, MPEG-1, H.262 (MPEG-2), H.263 and MPEG-4 [3, 4, 5, 6, 7], H.264 is a hybrid codec which combines blockwise transform coding and motion-compensated predictive coding to reduce the redundancy of a video signal. Overviews of modern video coding and in particular of H.264 can be found in [8, 9, 10, 11]. Two technically similar video coding standards, Microsoft's SMPTE VC-1 and the Chinese Advanced Video coding Standard AVS, are presented in [12, 13] and in [14]. Compared to H.263, H.264 achieves bit rate reductions of up to 50% at a comparable quality. This gain is the result of a combination of new features introduced in the standard: these include better motion-compensated prediction with multiple reference frames and varying block sizes down to 4x4 pixels [15], spatial prediction of intra-coded blocks, and improved entropy coding [16]. A reference software implementation of H.264 has been made freely available [17].

The standard specifies three major profiles: the *Baseline* profile which is mindful of the computational complexity, the *Main* profile, designed to take full advantage of the coding efficiency of H.264, and the *Extended* profile, which includes a number of enhancements for streaming applications [18]. Recent additions to the standard include an extension for higher fidelity (e.g., 10 bits/sample) video signals called FREXt (Fidelity Range Extension) [19] and an extension for Scalable Video Coding (SVC) [20, 21].

The basic components of a hybrid video encoder are shown in Fig. 2.1. 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 other frames ("motion-compensated predictor") or in the same frame ("intra predictor"). As in still picture coding, intra-prediction exploits correlation among adjacent pixels in the image. More specific to video is motion-compensated prediction that uses one or several previously encoded frames as references to predict the current frame. 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; Bi-directionally 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 and, since context-based entropy coding is used, such an error will affect all the following syntax elements until a re-synchronization 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.

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

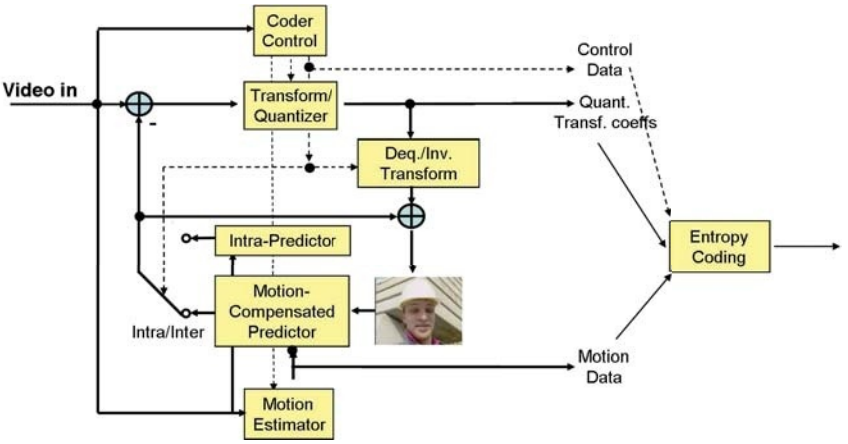

Fig. 2.1. Diagram of a motion-compensated hybrid video coder according to H.261, MPEG-1, MPEG-2, H.263, MPEG-4, or H.264/AVC standards. The intra-inter switch controls whether spatial or temporal prediction is used for compression. Dependency between frames is introduced via the motion-compensated inter-frame prediction when P frames and B frames are encoded.

2.1.2 Distortion Models

Performance of Motion-Compensated Video Coding

To study the performance of hybrid video coding, a theoretical framework is developed in [22]. An analysis of the rate-distortion efficiency of motion-compensated coding is presented in this paper, where a closed-form expression is obtained by assuming the different image signals and motion-compensated predictors are stationary and jointly Gaussian zero-mean signals. Hence, the resulting rate-distortion function can be thought of as an upper bound to the rate-distortion function for a non-Gaussian image signal with the same power spectral density. Although this is a simplification, the model has been widely used in the literature to evaluate the performance of several image or video encoders. The performance of I frames and P frames is derived for integer-pixel and fractional-pixel motion accuracy in [22] and studied further in [23]. The rate-distortion efficiency of B frames can be obtained from the model extension to multi-hypothesis predictive coding proposed in [24]. Other extensions are presented in [25, 26, 27] where the effect of the size of the set of predictors and of the correlation between the different predictors is analyzed. In [28] and [29], the authors show that the model can also be helpful to quantify the performance of scalable video codecs.

Empirical Distortion Models

The model described in [22] is general and well-suited to gain insight on the influence of different elements composing a video coding system. However