ROBUST VIDEO STREAMING OVER TIME-VARYING WIRELESS NETWORKS

A Thesis
Presented to
The Academic Faculty

by

Mehmet Umut Demircin

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology
August 2008

ROBUST VIDEO STREAMING OVER TIME-VARYING WIRELESS NETWORKS

Approved by:

Professor Yucel Altunbasak, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Chuanyi Ji
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Ghassan AlRegib
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Ozlem Ergun
School of Industrial and Systems Engineering
Georgia Institute of Technology

Professor Russell M. Mersereau
School of Electrical and Computer Engineering
Georgia Institute of Technology

Date Approved: June 4, 2008

To my family,

Hicabi, Kezban and Ahmet Oguz.

ACKNOWLEDGEMENTS

First, I would like to thank my advisor, Prof. Yucel Altunbasak for his great guidance throughout my Ph.D. study. He was a very good mentor, colleague and friend over these 6 years. I will always need his wisdom and expertise in my career.

I am grateful to have a very distinguished thesis committee: Prof. Russell M. Mersereau, Prof. Ghassan AlRegib, Prof. Chuanyi Ji and Prof. Ozlem Ergun. I thank them all for being very supportive of my research and for their constructive critiques that have greatly contributed to my thesis.

I would like to thank the members of the Multimedia Computing and Communications Lab, as well as the members of the Center for Signal and Image Processing for their support and friendship.

Dr. Ali Cengiz Begen has been my loyal friend and colleague for over 17 years. He was the one who persuaded me to apply for Ph.D. at Georgia Tech. I will always be grateful for his support through the rough times of the life.

I also want to thank Tarik Arici for friendship and support during the Ph.D. years and specifically during the thesis writing stage.

During summers of 2004 and 2005, I interned in Sharp Laboratories of America. I would like to thank my mentor Dr. Peter van Beek, and manager Dr. Ibrahim Sezan. The research during the internship and our collaboration afterwards, with their funding, shaped majority of my thesis and resulted in many publications and patents.

In summer 2007, I interned in DoCoMo USA Labs under the supervision of Dr. Ulas Kozat. The last chapter or my thesis is based on the work performed with collaboration with Dr. Ulas Kozat, Dr. Oztan Harmanci, Dr. Sandeep Kanumuri,

and Prof. Reha Civanlar. I would like to thank them all for the wonderful opportunity.

I want to thank my country, Turkey, for the education and opportunities she provided me. I would like to thank all my teachers and professors. I specifically want to thank my primary school teacher Ayse Demiroren for the example she set and her guidance.

Most importantly, I want to thank my dad, mom, brother, grandmother and grandfather. Their love, sacrifice and support made me who I am. It is to them that I dedicate this work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Multimedia services and applications became the driving force in the development and widespread deployment of wireless broadband access technologies and high speed local area networks. Mobile phone service providers are offering wide range of multimedia applications over high speed wireless data networks. People can watch live TV, stream on-demand video clips and place videotelephony calls using multimedia capable mobile devices. Mobile devices will soon support capturing and displaying high definition video. Similar evolution is also occurring in the local area domain. The video receiver or storage devices were conventionally connected to display devices using cables. By using wireless local area networking (WLAN) technologies, convenient and cable-free connectivity can be achieved. Media over wireless home networks prevents the cable mess and provides mobility to portable TVs.

However, there still exit challenges for improving the quality-of-service (QoS) of multimedia applications. Conventional service architectures, network structures and protocols lack to provide a robust distribution medium since most of them are not designed considering the high data rate and real-time transmission requirements of digital video.

In this thesis the challenges of wireless video streaming will be addressed in two main categories. *Streaming protocol* level issues constitute the first category. We will refer to the collection of network protocols that enable transmitting digital compressed video from a source to a receiver as the streaming protocol. The objective of streaming protocol solutions is the high quality video transfer between two networked devices.

Novel application-layer video bit-rate adaptation methods are designed for handling short- and long-term bandwidth variations of the wireless local area network (WLAN) links. Both transrating and scalable video coding techniques are used to generate video bit-rate flexibility. Another contribution of this thesis study is an error control method that dynamically adjusts the forward error correction (FEC) rate based on channel bit-error rate (BER) estimation and video coding structure.

The second category is the *streaming service* level issues, which generally surface in large scale systems. Service system solutions target to achieve system scalability and provide low cost / high quality service to consumers. Peer-to-peer assisted video streaming technologies are developed to reduce the load of video servers. Novel video file segment caching strategies are proposed for more efficient peer-to-peer collaboration.

CHAPTER I

INTRODUCTION

Multimedia services and applications became the driving force in the development and widespread deployment of wireless broadband access technologies and high speed local area networks. Mobile phone service providers are now offering a wide range of multimedia applications over high speed wireless data networks. People can watch live TV, stream on-demand video clips and place videotelephony calls using multimedia capable mobile devices. The variety and quality of these applications are increasing every day. Mobile devices will soon support capturing and displaying high definition video. Similar evolution is also occurring in the local area domain. The video receiver or storage devices were conventionally connected to display devices using cables. By using wireless local area networking (WLAN) technologies, convenient and cable-free connectivity can be achieved. Media over wireless home networks prevents the cable mess and provides mobility to portable TVs.

However, there still exits challenges for improving the quality-of-service (QoS) of multimedia applications. Conventional service architectures, network structures and protocols lack to provide a robust distribution medium since most of them are not designed considering the high data rate and real-time transmission requirements of digital video.

1.1 Problems in Wireless Video Streaming and Relation to the State-of-the-Art

Concurrent transmission and display of audio-visual (AV) content is often referred to as *streaming*. Streaming eliminates the initial waiting time before video playback starts and the requirement for storing the entire video file as opposed to download

and play schemes. The size of streaming systems can range from single sender single receiver setup to a video-on-demand (VoD) service with thousands or millions of users streaming video concurrently.

We will classify wireless video streaming problems and solutions in two main categories. *Streaming protocol* level issues constitute the first category. We will refer the collection of network protocols that enable transmitting digital compressed video from a source to a receiver as the streaming protocol. The objective of streaming protocol solutions is the high quality transfer between two networked devices.

The second category is the *streaming service* level issues, which generally surface in large scale systems. Service system solutions target to achieve system scalability and provide low cost / high quality service to consumers.

1.1.1 Video Streaming Protocol Level Problems

The fast viewing advantage of streaming comes with the price of sensitivity to network transmission errors and throughput fluctuations. These network impairments may cause distortion in AV presentation in the form of frame freezes or defects, unless they are handled. Wireless links are more susceptible to quality inconsistencies in contrast to wired connections. User mobility, environmental changes, and interference from other electromagnetic signal sources cause channel capacity variation. The shared nature of the communication medium among multiple applications and users is another challenge for streaming protocols over both wired and wireless networks.

Two points of view are provided to streaming protocol level problems. First, video streaming problem is approached from a purely application layer perspective. This methodology fits to the scenarios where error control mechanisms are already deployed on the network, and the application does not have control over them. In such cases, tackling fluctuating application throughput becomes more important than recovering errors. We refer to this problem as application layer video transmission

rate control, and demonstrate its use in wireless local area networks in Section 1.1.1.1

Secondly, we look from a lower network layer perspective, and try to create a better transmission channel for video streaming. In this cross-layer optimization effort we developed error-control methods tailored for video streaming over wireless wide area networks. The developed forward error correction (FEC) scheme considers the coding structure and real-time requirements of the media stream for efficient error recovery. In Section 1.1.1.2, we provide a detailed analysis and a literature survey for this problem.

1.1.1.1 Application Layer Video Transmission Rate Control Problem

Widely used IEEE 802.11 wireless local area networking (WLAN) technologies provide cost-effective and convenient solutions for interconnecting home video sources and display devices. In addition to offering free mobility for portable displays, wireless media streaming eliminates the need for excessive audio-video (AV) cabling. High-quality live and stored video content from cable/satelite receivers, personal video recorders (PVR), DVD players and PCs can be distributed digitally to all wireless capable displays.

The maximum physical data rate of the 802.11b technology (11 Mbps) is able to support MPEG-2 encoded standard-definition (SD) video transmission. 802.11a/g (54 Mbps) networks can be used to disseminate high-definition (HD) video content. Although the 802.11 standards provide high speeds, they cannot always maintain the channel consistency demanded by delay sensitive video streaming applications. For instance, some devices and technologies operating in the same unlicensed frequency band, such as cordless phones, neighboring WLANs and Bluetooth, may cause interference. Furthermore, mobility and environmental changes increase the physical (PHY) layer packet loss rate. Most PHY layer losses are recovered by medium access

control (MAC) layer retransmission based error-control methods. Multiple transmission retries at the MAC cause drops in application throughput. In addition, most WLAN base stations and adapters employ automatic PHY transmission rate selection schemes to cope with reduced signal-to-noise ratio, which again translates into bandwidth fluctuation. Other traffic flows sharing the same network resources may also cause throughput degradation if the MAC does not support quality-of-service (QoS) mechanisms. Although the new 802.11e MAC enhancements for QoS [3] are able to provide dedicated bandwidth for media traffic, it has not been widely deployed in current systems.

Most prior studies on the area of networked multimedia focus on tailoring error control methods for transmission over error-prone channels. Delay-constrained automatic repeat request (ARQ) based error recovery methods primarily intend to bound retransmissions considering the real-time requirements of the multimedia [8], [47], [29], [16], [45]. In [39] Li and van der Schaar studied a similar problem and proposed MAC retry limit adaptation and queuing methods for scalable video transmission over WLANs. This cross-layer technique results in good video streaming quality. However, it involves application layer interaction with the network and MAC layers. Therefore its use with standard of-the-counter WLAN equipment is limited.

Much research has been reported recently on rate-distortion optimized packet scheduling [11], [10], [43], [44]. The objective of these methods is to detect losses using stochastic models and determine the transmission/retransmission order of the packets to minimize the expected distortion. Due to the fact that the MAC and PHY layers recover most errors, WLAN applications experience almost lossless but varying bandwidth, channels. Moreover, these methods use per-packet feedback messages, which are not preferred in WLANs, since the forward and backward flows should share the same resources. As a result, rate-distortion optimized scheduling cannot be applied as-is to our target system.

When the available wireless bandwidth drops below the video bitrate, packets tend to backlog at the serving station. New video packets handed to lower network layers by the server application are queued behind the backlogged packets, and therefore are subject to varying delay (equivalently varying bandwidth). Playout buffering is a commonly used technique for compensating for the delay jitter. However, the playout buffering duration is limited to small values (100 ms - 1 sec) since the initial waiting time reduces user satisfaction. If the bandwidth degradation persists, the playout buffer underruns and causes video frame freezes. In [57], Stockhammer et. al. provided an analysis to determine the minimum initial delay for a given video stream and a deterministic variable bit-rate (VBR) channel. The performance of application layer error resilience [66], [53] and concealment methods [49], [63] are very limited in bursty error events caused by buffer underflows.

An application layer method to prevent buffer under-runs is to reduce the AV bit-rate adaptively by estimating the future bandwidth. The source rate can be adjusted easily by dropping layers if the video is scalably coded. For instance, in a recent study [36], Kim and Ammar presented a scalable Internet video streaming strategy to cope with the throughput variations of the transport control protocol (TCP). Most of the popular video applications (e.g., DVD, PVR) do not employ scalable content. For these scenarios rate adaptation should rely on transrating. A real-time transrating operation is achieved by partially decoding the video bitstream and re-quantizing the transform coefficients.

In a related study [9], Cabrera et al presented a stochastic dynamic programming based rate-control method for wireless video transmission. Rate-control policies are pre-calculated for every channel and video source state in this method. A similar approach for burst-error channels was proposed by Hsu et al in [25].

The method we developed in this thesis work considers the delay requirements of the stream by estimating the playout buffer fullness. Frame freezes are prevented

and video quality is maximized with rate adaptation. Stricter separation between network layers, use of less frequent feedback messages and efficient management of real-time constraints differentiates our method from previous works.

1.1.1.2 Cross-Layer Optimized Error Control Problem

Mobile video is an important application class for wide area wireless systems. Wireless clients are able to stream video clips from servers located on the Internet or communicate via video capable portable devices. However, the heterogeneity of the physical environments and network architectures in the communication path degrade the performance [7, 67]. In this context, a multimedia gateway located at the interface between the wired and wireless domains may allow us to develop optimized cross-layer protocols for the wireless portion of the connection [20, 23].

The time-varying and noisy nature of the wireless channels give rise to bit errors and packet erasures [64, 21, 69]. Multimedia applications can tolerate only small data loss rates due to the compressed structure of the media. The distortion caused by the errors may propagate because of the predictive coding often used in the contemporary media encoding standards. Furthermore, each packetized media unit has a presentation deadline at the client, which is determined by the interactivity requirements and buffer limitations. The deadline constraint imposes restrictions on the transmission delay of video packets. Failing to deliver the unit by the deadline causes audio-visual quality degradation in the multimedia application. These requirements point out a need for the intelligent use of error control schemes in real-time media communication protocols in order to guarantee a certain quality-of-service.

Automatic Repeat Request (ARQ) and Forward Error Correction (FEC) are two commonly used techniques for error recovery. In ARQ schemes, the sender transmits a packet and waits for an acknowledgement from the receiver. The packet is retransmitted if a negative acknowledgement is received or no acknowledgement is received

until a pre-determined time. In FEC schemes, the sender incorporates parity and redundancy into the packets so that receivers can detect and repair corruptions and losses. Modern wireless systems such as 2.5G/3G often use hybrid combinations of these two methods. Our initial study focuses on FEC based methods.

The error correction capability of an FEC code increases with the amount of redundancy incorporated. Using a fixed redundancy rate results in under-utilization of the capacity since the radio channel bit-error-rate (BER) fluctuates over time. An FEC code with a fixed rate would not only be unable to correct errors at high BERs, but also may incorporate unnecessary redundancy if the channel is clear. This observation leads to the design of channel-adaptive coding schemes, in which the sender chooses the proper coding rate based on BER estimation.

The FEC code rate selection problem is generally independent of the source characteristics for classical data applications. However, error recovery may take the unequal packet importance and real-time requirements of the media into account for the multimedia streaming applications. Due to the limited bandwidth and the delay requirements, the amount of channel resources (e.g., FEC redundancy, number of retransmissions, quality of the media content) spent for transmitting a packet affects the residual resources for subsequent packets. Preserving network resources for the more important subsequent video packets and for those packets that may face noisier channel conditions can be a better strategy. This strategy cannot be achieved by a channel allocation that does not take possible future events into account. Motivated by these observations, we argue that video quality can be increased by jointly optimizing FEC rate decisions for the current and subsequent video packets. The designed cross-protocol optimized error control solution can be deployed at the gateway, which is located at the wireless edge.

Our work differs from the previous studies in the literature mainly by the consideration of residual network resources for subsequent packets and proposed joint optimization. The most closely related work in terms of modeling the time-varying wireless channel and FEC rate adaptation is proposed by Elaoud and Parameswaran [19]. In this study, transmission decisions are made depending on the deadline and air-interface status. However, they do not incorporate the packet dependencies and the effect of future transmissions in their models. In the work by Kang and Zakhor [35], scheduling of video packets are done by adjusting deadline thresholds based on the importance determined by the position of frames inside the group of pictures and motion-texture content.

Chou and Miao [11] formulated and solved the problem of streaming packetized media over lossy packet networks in a rate-distortion optimized way. In [11] they proposed an algorithm called Iterative Sensitivity Adjustment (ISA) that is based on markov decision process to optimize the transmission/retransmission order and schedule by meeting rate constraints. Another closely related work is done by Chakareski and Chou [10]. They considered the problem of streaming packetized media over a lossy packet network through a base station to a wireless client. They proposed a streaming system based on a hybrid-II ARQ error control scheme also known as incremental redundancy (IR) transmission. Optimal use of the IR transmission scheme is determined by ISA algorithm.

Liu and Zarki [41] proposed a concatenated hybrid ARQ scheme that combines the advantages of both type-I and type-II hybrid ARQ schemes for low-bitrate video transmission over wireless channels. Wireless channels are modelled with multi-state Markov chains to model errors in the packet level. Kumwilaisak [37] explored a dynamic programming solution in coordinating the concatenated FEC code consisting of the Reed-Solomon (RS) code and the rate-compatible punctured convolutional (RCPC) code considering priority of each source packet and estimated instantaneous

channel condition. Qiao and Shin [51] presented a hybrid ARQ scheme for transmitting H.263 video sequences based on channel condition and delivery deadline constraints. They use an algorithm that adaptively selects the FEC code rate from a pre-determined code table in order to satisfy some quality constraints. None of the aforementioned methods jointly optimize the FEC code rate of current and future packets.

1.1.2 Video Streaming Service Level Problems

Video streaming is one of the most challenging services to offer because of the high and consistent bandwidth requirements of the digital video bitstreams. Furthermore, in large scale systems thousands or millions of users concurrently stream video. Service providers generally use a farm of video servers and lease very high speed data lines to accommodate large number of users that demand high quality video. Furthermore, the demand for video content shows time varying behavior. Users may rush the system at particular days of the week, for instance when a movie is newly released, or at particular hours of the day. The system should be designed to work robustly at the worst case scenarios. As the server and network costs increase the price of the video streaming applications becomes more expensive. It is crucial for service providers to develop system level cost reduction techniques to reach broader consumer base.

In conventional client-server based data services, a separate connection is opened for each client and data is unicasted to each of them. The number of simultaneous clients served by the system is limited by the server's storage disk performance and upstream bandwidth. Multicasting [52], [12] architecture offers a solution for this problem by enabling the replication of data at intermediate nodes of the network path. Therefore, the server does not have to send multiple copies of the same data destined to different nodes. Live video content such as TV programming can be distributed to many viewers over multicast trees without increasing the cost of

video server [40], [58]. However, multicasting capable network routers are not widely deployed over the Internet and wireless cellular data networks.

In video on-demand (VoD) services users start viewing the video at different time instants, therefore stream different parts of the video content. It is hard to take advantage of multicasting based schemes in VoD because of this fact. Near on-demand solutions are proposed in the literature to reduce the load of the VoD servers [26], [14]. Skyscraper broadcasting [27], harmonic broadcasting [33] and pyramid broadcasting [62] can be listed as the popular techniques adopting this approach. Most of these schemes involve periodic start times or user grouping to utilize multicasting. Therefore, users should wait for an initial time before starting to view the video. The waiting time is a negative factor that deteriorates the quality of the service perceived by the users hence it should be avoided as much as possible. Moreover, near on-demand solutions can not provide full fast-forward or rewinding functionality.

End-system or peer-to-peer multicasting is an application layer alternative to IP layer multicasting [28]. Users in such a system act similar to the multicast routers and forward the data they have received to the other peers, which are also viewing the same video. Asymmetry of downstream and upstream bandwidths in most wireless access technologies, such as 3G, limit the maximum throughput that can be achieved by end-system multicasting. Furthermore, the users participating in an end-system multicasting system may leave the system any time they want, which creates time-varying reliability for the streaming service.

In the thesis work we focus on a hybrid architecture where end-systems assist central servers in video distribution. Peer-to-peer (P2P) network refers to an overlay structure where hosts are able to exchange information among themselves without the need of a central server [5]. It is an alternative to the client-server paradigm commonly used in most web services. File sharing service over the P2P networks became one of the most dominating components of the Internet traffic. Even though

the P2P has a bad reputation of facilitating illegal file downloading, it also has a potential for assisting legitimate content distribution to reduce the cost. P2P collaboration can be useful in these applications by utilizing the unused upload bandwidth of the end users. In fact, popular Bittorrent [1] P2P file sharing application is commonly used to distribute the new Linux operating system releases. Use of P2P data distribution techniques for streaming live and on-demand video [38], [59], [34], [15], [68], [71], [18], [30],[42], [55], [32] became popular with the availability of high-speed network access technologies.

1.2 Organization and Contributions of the Thesis

The thesis is organized as a series of chapters. Separate research problems are discussed in each chapter. Motivation of the problem and related work are provided in the introduction sections. Proposed techniques are discussed and experimental results are presented in the body of chapters. Conclusion section of chapters summarize the findings.

The outline of the thesis and contributions are as follows:

Chapter 2 explores an application layer streaming protocol improvement that enhances high quality video streaming over wireless home networks. Video rate is dynamically adjusted via transrating to adapt to time-varying wireless bandwidth. A novel technique called *"Delay-constrained and R-D optimized transrating"* is developed. Experimental results are provided to demonstrate the achieved video streaming quality improvement.

Chapter 3 studies the use of scalable coded video for rate-adaptive video streaming over wireless local area networks. New features of the emerging H.264/SVC video coding standard are utilized to develop innovative streaming methods. Results from realistic simulations are presented.

Chapter 4 investigates cross-network layer collaboration techniques for improving

real-time multimedia streaming over wireless wide area networks. Forward error correction (FEC) based error protection methods are optimized using video bitstream structure and end-to-end latency constraints. A technique named as *"Finite-Horizon FEC-Rate Adaptation"* is developed.

Chapter 5 explores streaming service level solutions to reduce the cost of building large scale video-on-demand platforms. Peer-to-peer assisted video streaming technologies are developed to reduce the load of video servers. Novel video file segment caching strategies are proposed for more efficient peer collaboration. Computer simulation models are constructed to test techniques at diverse set of scenarios.

CHAPTER II

DELAY-CONSTRAINED AND R-D OPTIMIZED TRANSRATING FOR HIGH-DEFINITION VIDEO STREAMING OVER WLANS

2.1 Introduction

This study targets maximizing the quality of video transmission over wireless home networks. Source rate-adaption methods based on video transcoding are proposed to cope with variable channel bandwidth.

In our target platform, real-time bandwidth estimation is performed using video packet inter-arrival time measurements at the receiver. Therefore, no extra probing traffic is introduced. Periodic feedback messages are sent to the server after every measurement.

A video rate-adaptation method that solely relies on estimating average bandwidth is useful for tracking the long term variations. However, wireless channels also pose short term fluctuations which cannot be foreseen. Because of these unforeseen variations, this method may fail to deliver video packets on-time, especially if the initial playout buffering duration is small. To address this shortcoming, we propose new methods that consider the delay requirements of the stream by estimating the playout buffer fullness. Results of an initial version of this method were presented in [61] and [60]. This delay-constrained rate adaptation algorithm chooses the maximum possible rate for a picture frame while making sure it is fully received by its decoding deadline. This approach significantly reduces the number of playout buffer under-run events, compared to bandwidth-adaptive non-delay constrained methods.

We further generalized and improved the previously mentioned method by extending the delay-constraint concept to multiple picture frames, instead of one. Multi-frame constraints enable us to select various time scales for rate adjustments. We aim to keep the playout buffer fullness over a minimum level on the selected time scale. This novel feature of our method avoids unnecessary rate reductions and provides flexibility for assigning rates to frames depending on their coding types, hence reduces the transrating distortion.

The rest of this chapter is structured as follows: In Section 2.2 we will describe the used WLAN AV streaming system and summarize network monitoring and feedback mechanisms. The delay constrained rate adaptation method is explained in Section 2.3. This section also includes the transrater efficiency optimization and an automated time-scale selection technique. Experimental setup and comparison of different rate-adaptation methods are presented in Section 2.4.

2.2 Audio-Video (AV) Transmission System

The WLAN media streaming system envisioned is depicted in Figure 1. AV sources connected to the media-server/transrater and the WLAN access point, form the sender side of the system. The media-server acts as a gateway that adjusts the bitrate of the input bitstream and sends the packetized media over the access point. In this study, we concentrated on MPEG-2 video, since it is the widely used standard in digital TV and DVDs. We particularly tested the transmission of high-definition (HD) quality video, which poses the most challenging scenario because of its high bitrate ($\sim$16.9 Mbps).

A simple and fast open loop transrating technique that partially decodes the bitstream onto the quantization stage and performs re-quantization is utilized. The increased level of quantization results in higher compression rates at the expense of reduced image detail. The main drawback of the open loop transrating is the

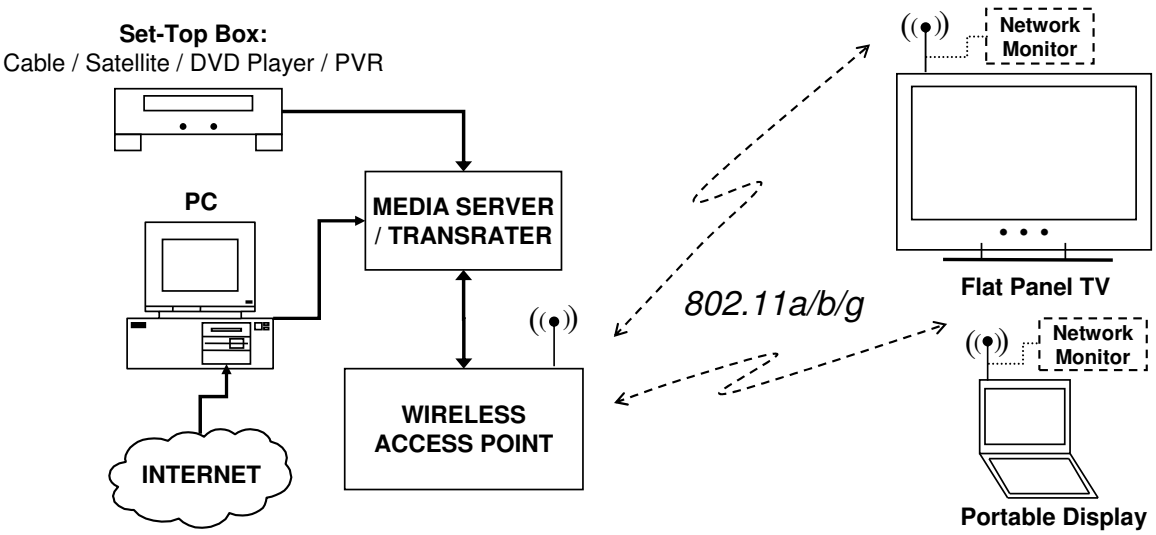

Figure 1: WLAN audio-video (AV) transmission system

drift error problem caused by uncorrected inter-frame prediction residuals after re-quantization. Closed loop transraters solve this problem by introducing error feedback mechanisms. However, additional processing power demanded by real-time closed loop transraters makes their deployment uneconomical for the home video streaming systems.

Display devices equipped with WLAN interfaces and video decoders form the client side of the system shown in Figure 1. Another role of the client is to monitor the network statistics and send feedback messages.

User datagram protocol (UDP) is used at the transport layer on top of the 802.11a/b/g WLAN technologies. The retransmission mechanism of TCP is not required, since the MAC and PHY layers recover most of the errors. In addition, the throughput variation due to congestion control is prevented by avoiding TCP. Note that the transmission medium is not shared with other data flows.

2.2.1 Wireless Local Area Network (WLAN) Link Monitoring

The effectiveness of rate adaptation depends on accurate and timely inference of the network state. To update the AV server, we employed feedback messages and a network monitoring module at the client side of the system. Our bandwidth estimation method is based on measuring the inter-interval times of sequentially (bursty) transmitted packets, as illustrated in Figure 2. AV bitstream packets are used for this purpose without inserting extra probing traffic. In the literature, similar techniques, also called packet train dispersion methods, are used to probe end-to-end Internet bandwidth [50], [31], [48].

The media server of the proposed system packetizes each video frame into fixed size packets and forwards them to the UDP layer as a burst. We selected the packet size as 1500 bytes (IP packet size) in order to minimize the overheads due to WLAN headers. Since the bitrate of the HD video considered in this study is high, about 16.9 Mbps, video frames are fragmented into multiple packets. As a result, packet trains (burst of video packets) are periodically transmitted at every frame interval.

Inter-arrival times of the packets in the train may vary because of the MAC layer loss recoveries and PHY layer transmission rate adjustments. Figure 2 presents an example illustration of this phenomenon in ideal and erroneous channel conditions. In ideal channel conditions, packets are received with equal intervals, which also represents the maximum achievable bandwidth. Error-prone wireless channels cause MAC layer losses, marked with stars in Figure 2, and retransmissions. Time slots with longer durations indicate packets transmitted at a lower PHY transmit rate. Inter-arrival times observed at the application layer increases as the channel quality deteriorates. Moreover, frames are subject to queuing delay if the packets are backlogged at the MAC.

We calculate the throughput observed during the transmission of a video frame using the time difference between the last and first packet receive events (Δt). Since

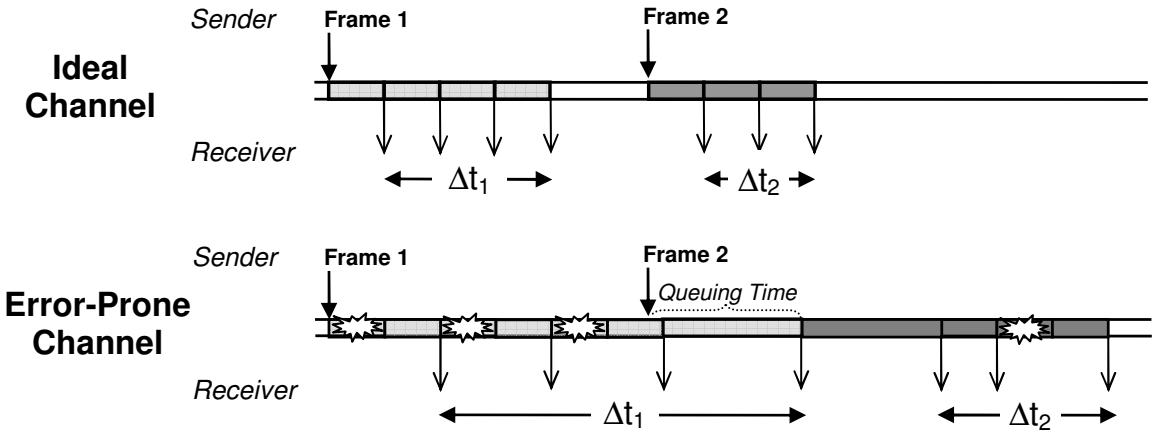

Figure 2: Video packet inter-arrival time measurements

the channel never stays idle during this interval, throughput is actually equal to the link bandwidth.

$$H = \frac{P \times (M-1)}{\Delta t} \tag{1}$$

In this equation, H represents the measured bandwidth, P is the fixed packet size, and M is the number of packets forming the packet train (video frame).

Bandwidth measurements are subject to errors that may be considered as noise. For example, the limited resolution of the clock used to measure packet arrival times may cause such errors. To compute a final estimate of the bandwidth $\hat{H}_k$ at frame k's receive time, our current implementation employs simple first-order IIR filtering, as follows:

$$\hat{H}_k = (1-w) \times \hat{H}_{k-1} + w \times H_k, \tag{2}$$

where w is a smoothing parameter between 0 and 1. Since the measurements become more reliable as the packet train gets longer, w is adjusted to be proportional to the size of the burst (M) or duration (Δt). The final estimate of the bandwidth for a frame k, $\hat{H}_k$, is transmitted back to the sender immediately after receiving the last packet of the video frame.

The feedback messages also contain the sequence number of the last received packet/frame. When the video server receives this information, it may estimate the

total size of the backlogged packets at the MAC layer buffer ($\hat{B}$) using the sequence number differences. The backlog size at frame i's transmission time is estimated as:

$$\hat{B}_i = \sum_{j=m}^{i-1} v_j - 0.5.\hat{H}.t_c \quad (3)$$

In this calculation, m is the sequence number of the oldest unacknowledged picture frame and v_j is the size of frame j. The sum of unacknowledged frame sizes provides an initial estimate of the backlog. This initial estimate is further improved considering the fact that a portion of unacknowledged frame m may have been already received. t_c is the time elapsed after the actual transmission of frame m has started. Actual transmission starts immediately after the frame is forwarded to the lower layers if the channel is idle. The frame should wait for prior packets when the backlog size is greater than zero. t_c is multiplied with half of the bandwidth estimate to predict the received portion of frame m. A 0.5 factor is used to prevent under estimation of the backlog. Backlog size or duration is actually the dual of the playout buffer fullness. Playout buffer becomes smaller as the backlog size increases.

2.3 Problem Formalization and Solution Approach

The end-to-end transmission delay experienced by the video packets constantly grows if the link bandwidth drops below the video bitrate over a certain time interval. The undesirable results of this problem are long video frame freezes and jumps. The best solution for a continuous video viewing experience is source rate adaptation.

2.3.1 Bandwidth-Adaptive Non-Delay Constrained Rate Adaptation

A video rate-adaptation method can solely rely on bandwidth estimations. Such a method can track long term variations of the bandwidth and prevent backlog growth that causes network card buffer overflows. However, wireless channels also pose short term fluctuations which cannot be foreseen, as shown in Figure 3. In this two second

example, the actual bandwidth is sampled over 1/60 second intervals and the estimation using the method explained in Section 2.2.1 is plotted. Since the tolerable packet delivery delay is limited by the initial playout buffering duration, short term bandwidth variations create problems for this method.

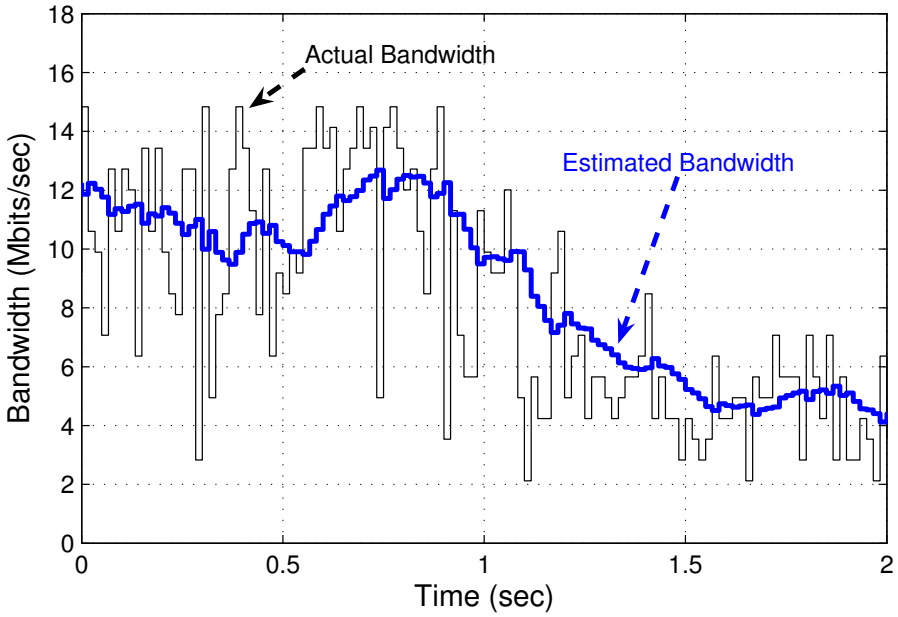

Figure 3: Fluctuating WLAN bandwidth and estimation

We implemented a bandwidth-adaptive non-delay constrained method that adjusts the rate of each video frame using the following formula:

$$x = min\left[1,\ 0.9 \times \frac{\hat{H}}{V}\right] \tag{4}$$

The transrating ratio, which is the fraction of a frame's transmission size to its initial size, is represented with x. $\hat{H}$ is the bandwidth estimate and V is the original bitrate of the video stream. The multiplier, 0.9, is used as a correction margin for bandwidth over-estimation errors.

We performed a 30 second long streaming simulation and plotted the cumulative delay histogram of the video frames in Figure 4. The delays of periodically transmitted video frames are measured as the time difference between their sending time and

receipt of their last packet. A detailed explanation of the test setup can be found in the experimental results section. In this experiment bandwidth is constantly below the video bitrate. Delay therefore constantly grows if the source rate is not adjusted. The first plot in Figure 4 demonstrates that the maximum observed delay is 0.4 seconds when the bandwidth-adaptive non-delay constrained rate adaptation method is used. On the other hand, if the initial buffer was 200 ms, only 65% of video frames could be delivered on time.

2.3.2 Single-Frame Delay-Constrained Rate Adaptation

A long initial buffering time prior to the start of the video presentation may seriously hurt user satisfaction. Hence, the bound on the delay should be reduced even further. We propose a delay-constrained rate adaptation method that takes the initial buffer duration into account and utilizes the backlog estimates. Backlog estimates help us in handling short term bandwidth variations. The first version of this method chooses the maximum possible rate for a picture frame while making sure it is fully received by its decoding deadline. The initial playout buffer duration is denoted by ΔT_E, and x_i and v_i represent the transrating ratio and original size of the current frame i. The single-frame delay constraint for the delivery of the frame i is expressed as follows:

$$\frac{x_i.v_i + \hat{B}_i}{\hat{H}} \leq \Omega.\Delta T_E \tag{5}$$

Maximum possible value of the transrating ratio is 1. The left term in this inequality is the delay estimate of frame i's last packet. We again use a safety factor, $0 < \Omega \leq 1$, against bandwidth and backlog estimation errors. This method is effective in preventing playout buffer underflows. The cumulative delay histogram (right plot of Figure 4) for a streaming experiment using 200ms initial buffer duration, shows that almost zero percent late packet rate is achieved.

In the next section, we first describe the reasoning behind extending the delay constraint from a single frame to multiple frames. Subsequently, the implementation

details of this novel method are discussed.

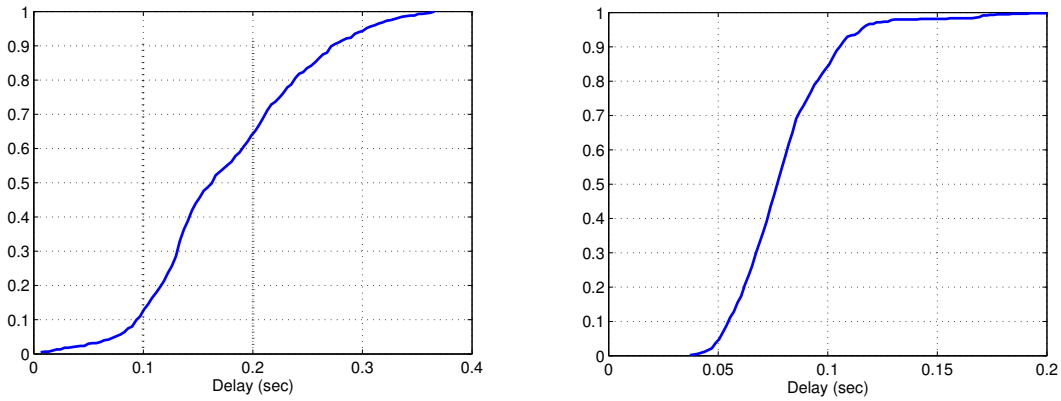

Figure 4: Cumulative histogram of frame transmission delays for bandwidth adaptive non-delay constrained (on the left) and single-frame delay-constrained (on the right) methods.

2.3.3 Multi-Frame Delay-Constraint Concept

While the primary objective of rate adaptation is to prevent playout buffer underflows, it is also desired to maintain a good video presentation quality level. The key for increasing the stream quality is to minimize the distortion caused by the transrating process. A few observations on the operation of the single-frame delay constrained rate adaptation lead us to a new method that achieves this goal.

Our first observation is that rate is not reduced in response to bandwidth drops as long as the current frame is delivered by its deadline, if the delay is constrained for a single frame. This behavior causes the playout buffer to vanish quickly, which in turn results in aggressive transrating when the backlog duration increases. Aggressive transrating, which corresponds to large degradation in frame quality, can be avoided especially when the initial buffering duration is long enough. We intend to solve this problem by increasing the rate adaptation time scale from a single frame to multiple frames. This extension lets us proactively manage the playout buffer size by gradually starting the bit-rate adjustment at an earlier stage. In Figures 5 and 6, we set the initial buffer as 500 ms, and plot a sample evolution of the playout buffer over

time, in the case of single- and multi-frame delay constraints. As demonstrated in Figure 5, the number of packets in the buffer fluctuates within a small range, which is an indicator of aggressive transrating. When the time scale is increased to 30 frames instead of one, initial buffering time is more efficiently managed as shown in Figure 6.

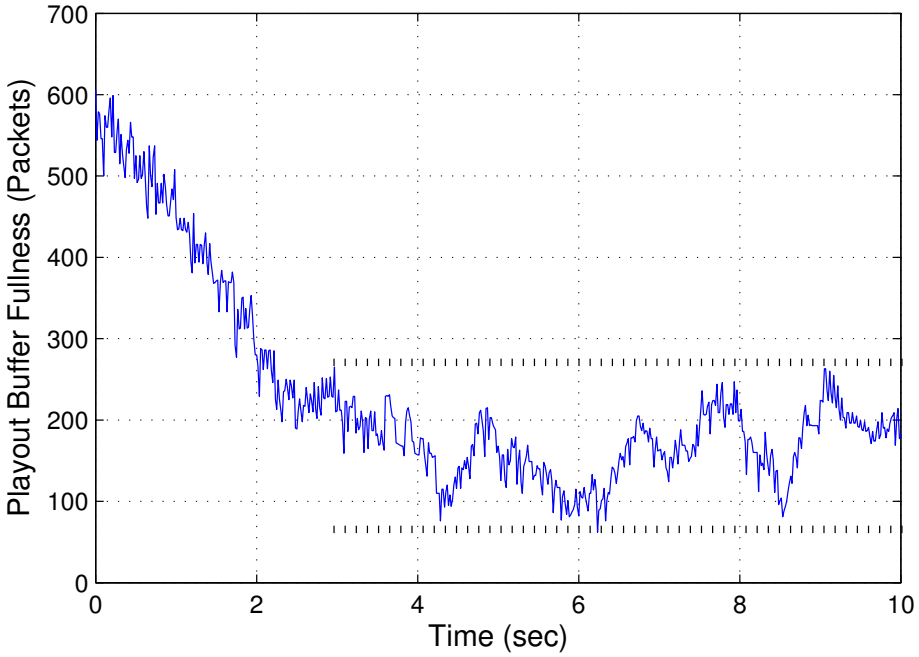

Figure 5: Playout buffer fullness - Single-frame delay-constrained rate adaptation

Varying sizes and different encoding types of the picture frames inside a compressed AV stream should be taken into account for minimizing the distortion. When the rate is constrained for a single frame, the transmission bit budget of large and more important I and P frames are computed the same way as it is done for the smaller and less important B frames. When the end-to-end bandwidth drops, the rate of I and P frames are reduced more than B frames because they are larger in byte size. By constraining the delay of multiple frames, we gain the power of distributing the cumulative rate freely over the frames in the projected time scale. The rate-distortion characteristics of frames are utilized in this rate allocation procedure.

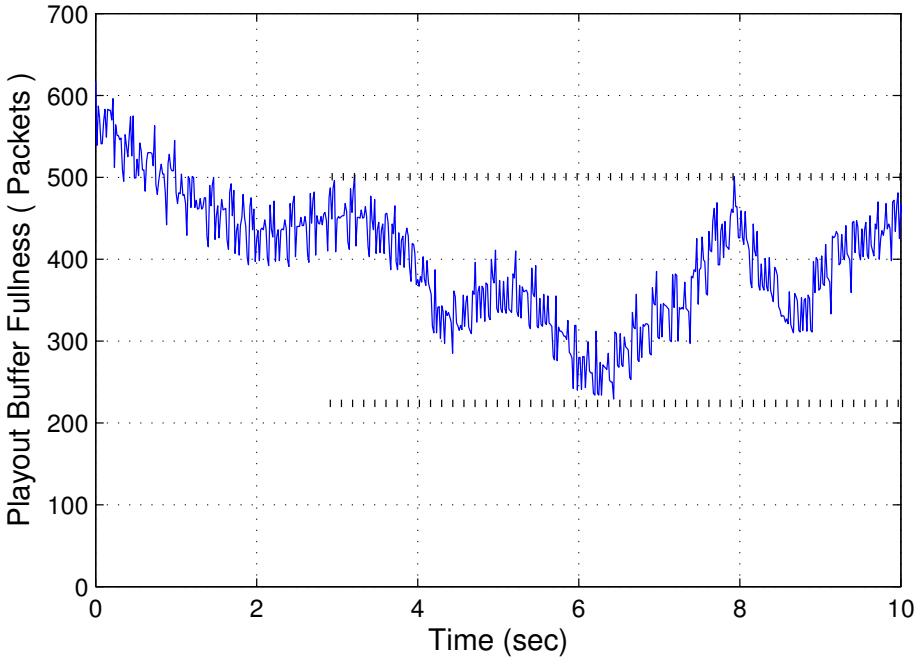

Figure 6: Playout buffer fullness - Time-scale and R-D optimized transrating

As a result, more rate is assigned to I and P frames as depicted in Figure 7.

In the following section, we describe how the transmission bit budget is computed over different time-scales.

2.3.4 Time-Scale Based Transmission Bit-Budget Computation

The rate adaptation method, located at the server, is responsible for determining whether, or how much, the current frame's size will be reduced. Figure 8 depicts how the rate adaptation method and the transrater are located in the overall WLAN streaming system. The first step of this new method is to calculate the total transmission bit-budget, R_G, for a group of multiple video frames in the projected time scale. Let us assume that the group consists of the current frame and h future frames, i.e. $\mathcal{G} = \{i, i+1, \ldots, i+h\}$. The time span of the frame group is equal to $h.\Delta T$ (ΔT is the inter-frame interval). Time scale gets larger as the number of frames in the

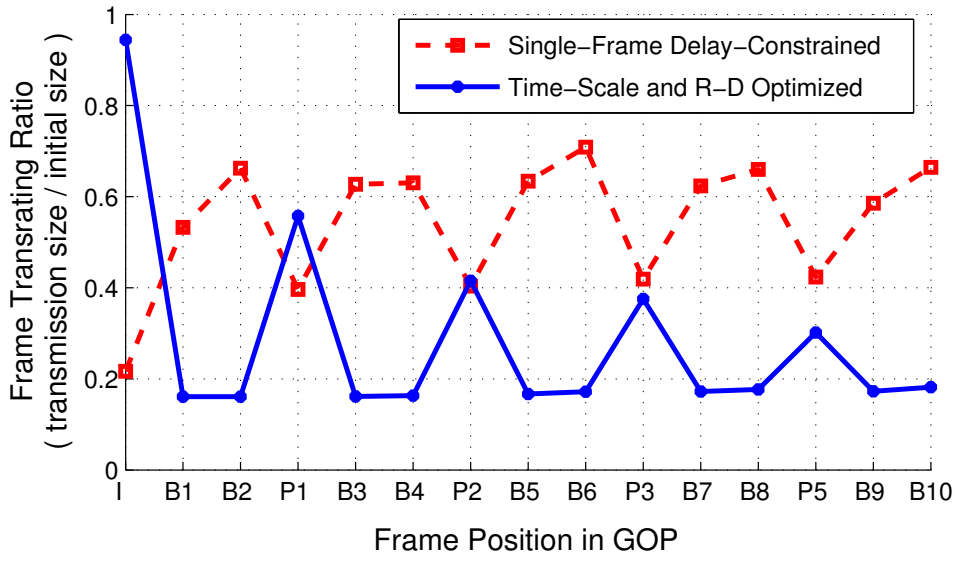

Figure 7: Average transrating ratios at specific GOP positions

group increases.

The total transmission bit-budget for the frame group is computed such that the last frame of the group is delivered to the client before a specified delivery time: $t_i + h.\Delta T + \Omega.\Delta T_E$. Similar to the single-frame delay-constrained method, i represent the sequence number of the current frame and t_i is the current time instant. In the new method, Ω (≤ 1) is a constant multiplier for expressing the maximum backlog time target in terms of ΔT_E (see Figure 8). Backlog target parameter also determines the minimum playout buffer fullness-ratio objective $(1 - \Omega)$.

The $\hat{H}.(h.\Delta T + \Omega.\Delta T_E)$ expression represents the estimated transmission capacity for the interval until the target deadline of the group's last frame. The total bit-budget for the new frames (in the group) is computed by subtracting the size of the packets that are already waiting (at the sender) from the capacity (defined above) :

$$R_G = \hat{H}.(h.\Delta T + \Omega.\Delta T_E) - \hat{B} \tag{6}$$

If the calculated total transmission bit-budget, R_G, is smaller than the total size of

the picture frames in the group, the rate of the video should be reduced in the time scale that corresponds to the frame group. In the next section we propose a technique that aims to allocate the total bit-budget efficiently among the frames of the group.

The values of group size, $h+1$, and the delay target parameter, Ω, that result in best streaming quality, depend on channel conditions, video bitrate, and initial buffer size. We provide an analysis and develop an automated parameter selection method in Section 2.3.6.

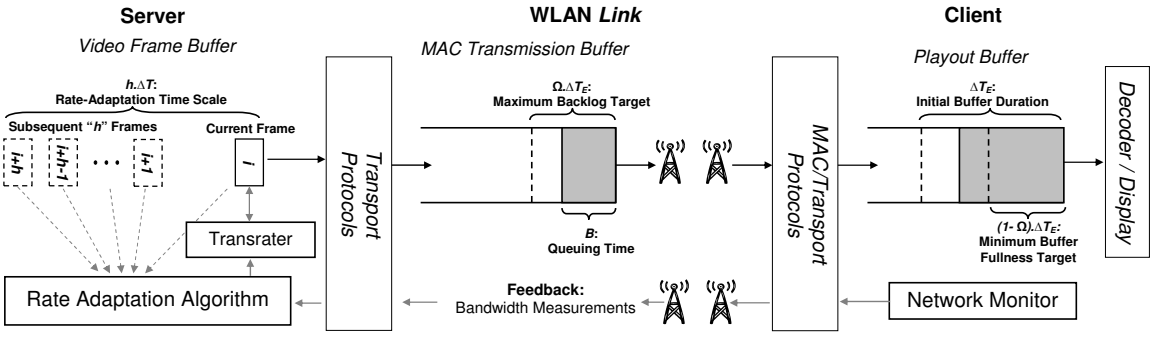

Figure 8: Block diagram of the rate adaptive WLAN AV streaming system.

2.3.5 Rate-Distortion Optimized Transrating

Objective of preventing buffer underflows by rate adaptation is attained by bounding the transmission bit-budget over a given time scale, using multiple frame delay constraints. The second goal is to minimize the quality loss due to transrating. We now propose a rate-distortion (R-D) optimized provisioning of the transmission budget considering the video frame types.

When the size of a picture frame is reduced via transrating, the quality of that frame and the frames that use that frame as prediction reference are degraded. It can be observed that the rate-distortion characteristics of these frames differ strongly based on their coding type (I, P or B), and position inside a GOP (Group of Pictures). For example, since all frames in the GOP depend on the I frame, reducing an I frame's rate causes distortion in all frames of the GOP. Bit rate reduction of a B frame affects

only one frame, the frame itself, because B frames are not used as reference for coding other frames. Furthermore, P frames near the beginning of a GOP may have a greater impact on the overall quality compared to P frames near the end of a GOP. In the next section, simple models are sought in order to differentiate frames.

2.3.5.1 Transrater R-D Model Extraction

We aim to express the overall distortion of the frame group as a function of the rate reduction ratios (x_j) of the individual frames. Determining such a function is a difficult task in general, as the dimensionality of the problem grows with the number of frames in the group. Furthermore, subsequent optimization of the distortion over a high-dimensional search space given by the allowable values of the rate reduction ratios can be computationally very complex and time consuming. We propose an approach that approximates the overall distortion value by considering the effect on the overall distortion caused by rate reduction of single frames at a time. Breaking down the problem allows the distortion-optimized solution to be computed with low computational complexity.

We performed experiments using a sample set of MPEG-2 high definition video sequences (HARBOUR, CREW, RAVEN), each encoded at 16.9 Mbps with 15-frame IBBP GOP structure, and tried to fit models. In these experiments we transrated only a single frame within each GOP. Figure 9 shows how the average luminance (Y) peak signal-to-noise ratio (PSNR) of the whole sequence changes as a function of overall rate decrease. By looking at the plots, we conclude that first order functions are accurate enough to characterize the PSNR reduction at high bitrates.

The following expression formulates the average GOP PSNR decrease by transrating a frame at a specific GOP position:

$$\Delta D_j(x_j) = a_{p(j)}.x_j + b_{p(j)}, \qquad \text{if } x_{min} < x_j < 1 \tag{7}$$

In Equation 7, $p(j)$ is a function that maps the frame sequence number, j, to

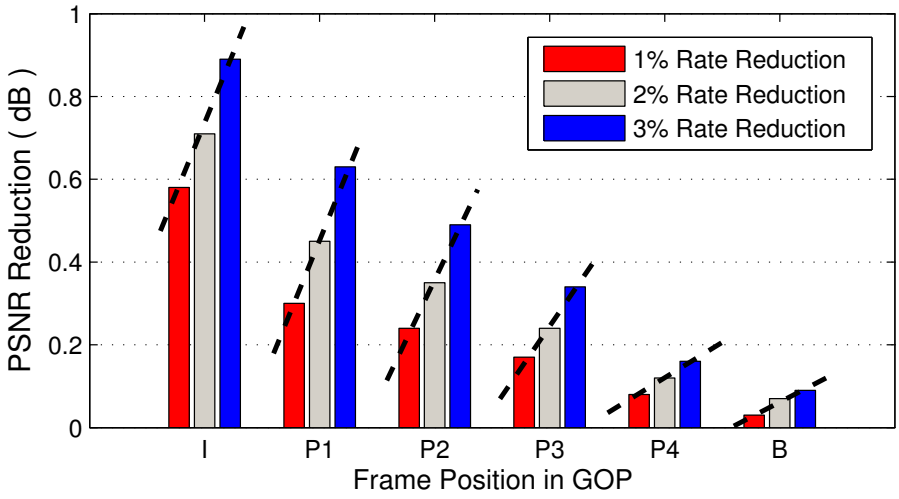

Figure 9: PSNR reduction due to transrating and linear model fitting (Sequence: HARBOUR)

the position in the GOP. a and b parameters are extracted by least-square fit to the data. Note that PSNR reduction is zero if the frame is not transrated, i.e. $\Delta D_j(1) = 0$, as an exception to the linear model. There is also a limit on the minimum transrating ratio, x_{min}, that can be achieved by re-quantization based transrating. Content independent, average rate-distortion model estimates can be produced by averaging the model parameters (a's and b's) over a large set of sequences. Table 1 shows the calculated parameters for three HD test sequences and their statistical average. The parameters are almost the same for the B frames throughout the GOP. As expected I frames have the biggest impact on the distortion.

2.3.5.2 Distortion Optimized Allocation of the Transmission Bit-Budget

Extracted rate-distortion models can be used for efficiently allocating the total transmission bit-budget. Quality degradation over time is balanced by minimizing the maximum of PSNR reductions among the frames in the group (Equation 8). This optimization target is achieved when the values of the distortion changes ΔD_j are

Table 1: Linear PSNR reduction model parameters

	HARBOUR		CREW		RAVEN		AVERAGE	
	a	**b**	**a**	**b**	**a**	**b**	**a**	**b**
I	-3.22	3.65	-0.94	1.19	-1.88	2.18	-2.01	2.34
B1	-0.13	0.13	-0.18	0.18	-0.13	0.13	-0.15	0.15
B2	-0.13	0.13	-0.19	0.19	-0.13	0.13	-0.15	0.15
P1	-1.57	1.74	-0.80	0.95	-1.32	1.48	-1.23	1.39
B3	-0.13	0.13	-0.18	0.18	-0.13	0.13	-0.14	0.15
B4	-0.13	0.13	-0.19	0.19	-0.13	0.13	-0.15	0.15
P2	-1.21	1.35	-0.69	0.83	-1.06	1.19	-0.99	1.12
B5	-0.13	0.13	-0.19	0.20	-0.13	0.13	-0.15	0.15
B6	-0.13	0.13	-0.17	0.17	-0.13	0.13	-0.14	0.14
P3	-0.82	0.92	-0.44	0.55	-0.74	0.83	-0.67	0.77
B7	-0.13	0.13	-0.17	0.18	-0.13	0.13	-0.14	0.15
B8	-0.12	0.12	-0.16	0.17	-0.13	0.13	-0.14	0.14
P4	-0.39	0.44	-0.20	0.27	-0.39	0.44	-0.33	0.38
B9	-0.13	0.13	-0.17	0.17	-0.13	0.13	-0.14	0.14
B10	-0.12	0.12	-0.16	0.16	-0.12	0.13	-0.13	0.14

equal. In addition, timely delivery is ensured with the transmission bit-budget constraint (Equation 9). The optimization problem is expressed as follows:

$$\text{minimize} \quad \max_{j \in G} \Delta D_j(x_j) \tag{8}$$

$$\text{s.t.} \quad \sum_{j \in G} r_j = \sum_{j \in G} x_j.v_j \leq R_G \tag{9}$$

$$x_{min} \leq x_j \leq 1.0 \quad \text{for} \;\; j \in G. \tag{10}$$

This problem can be solved, with very little computational power, using the linear models in Equation 7 in an iterative fashion to satisfy the lower (x_{min}: transrater limit) and upper bounds on the rate (Equation 10). The iterative algorithm is explained in Algorithm 1.

If the computed transmission size of the current frame (i) is less than the original size, it is transrated by the computed factor (x_i) and then transmitted over the wireless channel. Note that although the algorithm is capable of computing allocated bit rates (or rate reduction ratios x_j) for frames in the group following the current frame, the system does not actually need to utilize these computed bit rates. Instead,

Algorithm 1: Iterative Distortion-Optimized Transrater Rate Allocation

Data: Frame-group transmission bit-budget (R_G), estimated frame sizes (v_j), transrater R-D model parameters ($a_{p(j)}, b_{p(j)}$)

Result: Transrating ratio of the current frame (x_i)

1. Initialize $\mathcal{G} = \{i, i+1, \ldots, i+h\}$ as the set of frames in the group.

2. Compute x_j for all $j \in \mathcal{G}$, such that:

$a_{p(j)}.x_j + b_{p(j)} = a_{p(i)}.x_i + b_{p(i)}$, for all $j \in \mathcal{G} \backslash \{i\}$
and, $\sum_{j \in \mathcal{G}} x_j.v_j \leq R_G$

3. **If** $x_j \geq 1$ for all $j \in \mathcal{G}$

3.1. **Return** $x_i = 1$ and exit.

4. **If** $min_{\mathcal{G}}(x_j) \geq x_{min}$

4.1. **If** $max_{\mathcal{G}}(x_j) \leq 1$

4.1.1. **Return** x_i and exit.

4.2. **Else go to** step 6.

5. **Else if** $min_{\mathcal{G}}(x_j) \leq x_{min}$

5.1. Find the k such that $x_k \leq x_l$ for all $k, l \in \mathcal{G}$

5.2. **If** $k = i$

5.2.1. **Return** $x_i = x_{min}$ and exit

5.3. **Else if** $k \neq i$

5.3.1. Set $\mathcal{G} = \mathcal{G} - \{k\}$, $R_G = R_G - x_{min}.v_k$ and **go to** step 2.

6. **If** $max_{\mathcal{G}}(x_j) \geq 1$

6.1. **If** $x_j \geq 1$ for all $j \in \mathcal{G}$

6.1.1. **Return** $x_i = 1$ and exit

6.2. Find the k such $x_k \geq x_l$ that for all $l \in \mathcal{G}$

6.3. **If** $k = i$

6.3.1. **Return** $x_i = 1$ and exit

6.4. **Else if** $k \neq i$

6.4.1. Set $\mathcal{G} = \mathcal{G} - \{k\}$, $R_G = R_G - v_k$, and **go to** step 2.

the algorithm will be run again at the time of transcoding the next frame (frame $i+1$), and the system will be able to take into account new information (for example updated bandwidth and backlog size information), etc.

The original size of the future picture frames (v_j) may not be known exactly if the AV content is not stored at the server. In such a case, the values v_j for each GOP position may be estimated based on the averages of the previous frames that were at the same GOP position.

2.3.6 Rate-Adaptation Time-Scale Optimization

In the previous sections we explained how the rate adaptation works for a selected time-scale. The time-scale that results in best streaming quality may change depending on the initial buffering duration and channel conditions. We first provide an analysis and then propose an automated rate-adaptation parameter selection method to achieve the performance of the optimally set ones.

2.3.6.1 Analysis on Rate-Adaptation Time-Scale and Buffer Fullness Target

The rate-adaptation time-scale parameter, h, and playout buffer fullness target parameter, Ω, are the two variables of our method. We performed experiments to figure out the effect of h and Ω on the streaming quality. In order to test the channel condition dependency, we used four different WLAN traces. These traces are labeled as 'good','deteriorated','bad' and a combination 'hybrid'. The details of the test setup are described in the results section. Expected transrater PSNR loss and late packet delivery percentage for sample experiments are plotted in Figure 10 and Figure 11. In order to plot these curves we changed one parameter while keeping the other fixed. Using longer time scales, i.e. larger h, improves the transrater efficiency. However, it may cause buffer underflows if the initial buffering duration is small or the channel has high bandwidth variation. This effect can be seen in the left plots of Figure 10 and Figure 11, where the late packet delivery percentage may increase quickly if h is

increased beyond a threshold. This threshold increases when a smaller Ω is selected. Intuitively, we can argue that smaller h values should be preferred when the channel quality deteriorates or a short initial buffering duration is set. Ω acts as a safety parameter against bandwidth and backlog estimation errors. Timely frame delivery can be guaranteed by choosing small Ω, but this also increases the PSNR loss (see the right plots in Figure 10 and Figure 11). Note that these results indicate that h and Ω are coupled in such a way that a longer time scale requires smaller buffer fullness target parameter (Ω). The optimal parameter pair (indicated with stars in Figures 10 and 11) is selected such that the late packet delivery percentage is zero and PSNR loss is minimum. Table 2 presents the optimal parameter values of the RAVEN sequence at different channel conditions and delay tolerances. These parameter pairs also result in close to optimal performance for other video test sequences. The most aggressive parameters are observed for the 'hybrid' trace, since it shows the largest bandwidth variation.

It is not feasible to manually set h and Ω in real time streaming scenarios, hence we developed automated techniques in the next section based on this analysis.

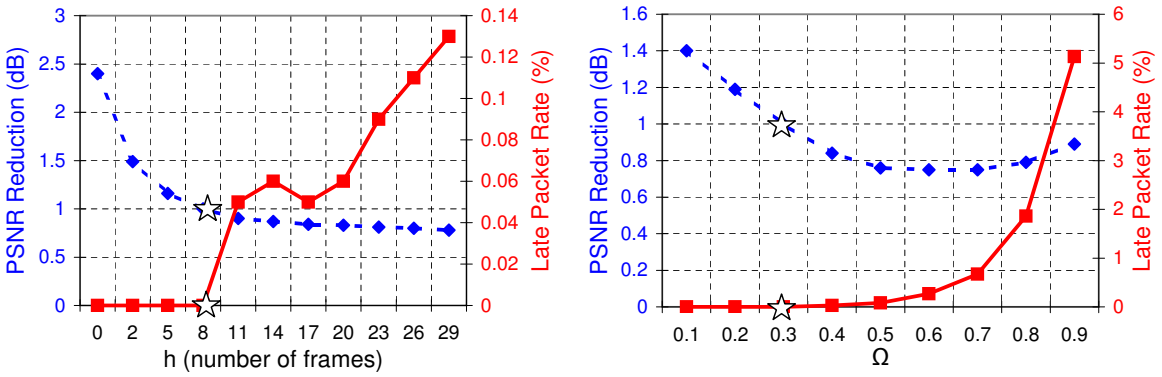

Figure 10: Effect of time-scale (left) and buffer fullness target (right) parameters on the streaming quality (Test sequence: RAVEN, Channel trace: 'Deteriorated', Initial playout buffer: 100 ms)

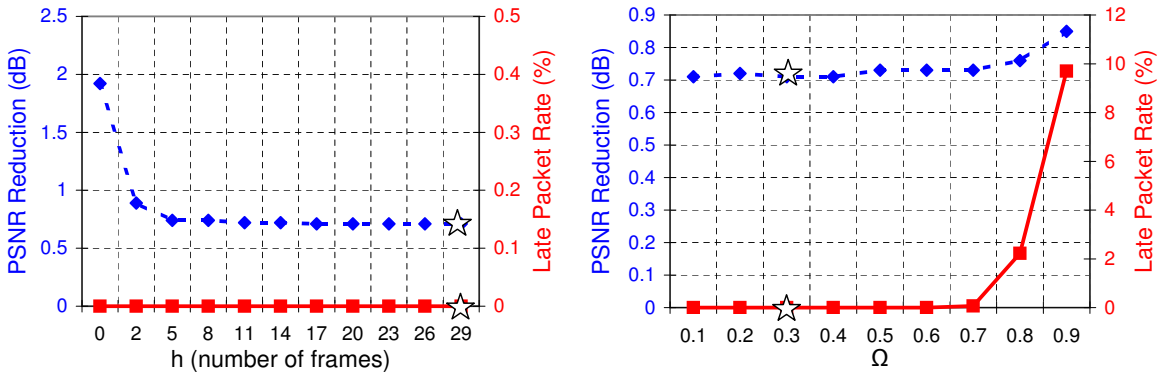

Figure 11: Effect of time-scale (left) and buffer fullness target (right) parameters on the streaming quality (Test sequence: RAVEN, Channel trace: 'Deteriorated', Initial playout buffer: 500 ms)

Table 2: Optimal rate adaptation parameters, Test sequence: RAVEN

Optimal Parameters		Initial Playout Buffer					
		100 ms		200ms		500 ms	
		h	Ω	h	Ω	h	Ω
Channel Conditions	Good	29	0.9	29	0.9	29	0.9
	Deteriorated	8	0.3	29	0.5	29	0.6
	Bad	2	0.1	20	0.1	29	0.3
	Hybrid	2	0.1	2	0.1	20	0.1

2.3.6.2 Automated On-line Parameter Selection Method

The objective of this method is to choose the frame group size $h+1$ and the sender backlog target multiplier Ω parameters in real-time, on a per frame basis. We expect the on-line parameter selection method to result in a streaming quality which is in most conditions close to the quality achieved by optimally selected static parameters.

We iteratively select the h parameter from a finite set of candidate integers, $\Phi = \{h_1, h_2, \ldots, h_n\}$. Smaller h values are useful to check short-term delay constraints; on the other hand, we use a larger h to investigate long-term delay constraints.

The elements of Φ can be selected as equally spaced integers between a minimum, h_{min}, and a maximum h_{max} value. In the sample implementations the minimum value is chosen as 0, which is equivalent to a single frame, and the maximum value is related to the video's GOP size, $(2 \times GOPSize - 1)$. The spacing between candidate values is selected equal to the period of the P frames. Therefore, $\Phi = \{0, 2, 5, 8, 11, 14, 17, 20, 23, 26, 29\}$ is used in the simulations.

This algorithm runs before the transmission of each frame. In Equation 11, the frame group transmission bit budget is computed for every candidate h value using the specifically calculated Ω_h. We will elaborate on the Ω_h calculation at the end of this section.

$$R_G^h = \hat{H}.(h.\Delta T + \Omega_h.\Delta T_E) - \hat{B} \tag{11}$$

Subsequently, the rate reduction ratio x_i^h for the current frame, i, is computed using the transmission bit budget. The h value that results in minimum rate reduction ratio is selected to determine the time scale. The purpose of such an approach is to satisfy both short term and long term delay constraints. Larger time scales are preferred when the playout buffer is full. On the other hand, our method aggressively responds to diminishing buffer, due to sharp bandwidth fluctuations, by reducing time-scale. The effective rate reduction ratio for the current frame is computed as:

$$x_i = \min_{h \in \Phi} x_i^h \tag{12}$$

The optimum value of Ω is coupled with the h parameter. Based on the performed analysis, we came up with an expression that maps the Ω to the initial buffering duration and h. The expression in Equation 13 captures the characteristics of Ω since it gets smaller as the buffer duration and time scale increase.

$$\Omega_h = \frac{\Delta T_E}{2\,(\Delta T_E + h.\Delta T)} \tag{13}$$

We compare the performance of this automated technique with the optimally selected parameters in the next section.

2.4 Experimentation and Results

2.4.1 Experimental Setup

We considered an experimental setup where the sender and receiver stations are connected over an ad-hoc wireless link. Background data traffic was not allowed during the video streaming session. The AV streaming system depicted in Figure 8 is implemented in a simulation environment using MATLAB. In order to emulate realistic WLAN channel conditions, we collected packet traces using two laptop computers equipped with D-Link DWL-AG660 (802.11a/b/g) network interface cards. Default settings of this card's windows operating system driver were used. Application layer traces were collected by continuously transmitting 1500 byte IP packets from the server to the client; thus the channel never stayed idle. Using packet traces allowed us to compare different streaming methods fairly at identical conditions. Three channel traces characterizing 'good' (average bandwidth is 20.26 Mbps), 'deteriorated' (14.07 Mbps), and 'bad' (7.78 Mbps) network conditions were collected. The duration of each trace was 30 seconds. A fourth trace, 'hybrid', was artificially generated by combining 10 second fragments from each of the previous three traces (Average bandwidth is 13.74 Mbps). As a result, the 'hybrid' trace included bandwidth variations over both short and long time-scales.

Our simulator was also provided with video frame traces. The MPEG-2 encoded HARBOUR (Y-PSNR: 35.27 dB), CREW (39.37 dB), and RAVEN (42.33 dB) test sequences at 1280x720 pixels (720p) resolution, 16.9 Mbps bitrate, and 60 Hz frame rate were used throughout the experiments. These 10 second sequences were looped three times to get more reliable results. The initial playout buffer duration (ΔT_E) was adjusted between 100 ms and 500 ms.

Our system utilizes a software MPEG-2 bit rate reducing transrater, with open-loop requantizing architecture. Late and lost packets were applied to the transrated video at the MPEG slice level, i.e., an entire slice is removed if it overlaps with a

late/lost packet. The quality of the final decoded output video was evaluated in terms of the PSNR and by visual comparison.

2.4.2 Comparison of the Rate-Adaptation Methods

We compared four different video rate adaptation methods at various WLAN channel conditions. The first method, which is labeled as "bandwidth-adaptive non-delay constrained", is the rate adaptation algorithm based on on-line bandwidth measurements, but it does not consider the delivery deadlines. The operation of this method is explained in Section 2.3.1 . Transrating ratio is selected using Equation 4.

The second method is the "single-frame delay-constrained" rate adaptation technique. The third method "static time-scale and R-D optimized" transrating uses the optimally selected h and Ω parameters. This method is included as a benchmark for the "automated time-scale and R-D optimized" technique.

A summary of the simulation results at four different channel conditions is presented in Tables 3- 6. The advantage of the delay-constrained (single-frame) rate adaptation over the bandwidth-adaptive non-delay constrained method becomes visible when channel conditions deteriorate. Short-term bandwidth variations variations are efficiently addressed by this method. The static time-scale and R-D optimized technique clearly outperform the bandwidth-adaptive non-delay constrained method and improves the quality up to 8.12 dB PSNR for the CREW sequence at bad channel conditions. The improvement achieved by generalizing the rate adaptation time-scale gets more significant as the initial playout buffer size increases. Time-scale and R-D optimized rate adaptation provides up to 4.42 dB gain compared to the single-frame delay constraint for the RAVEN sequence at the bad channel condition and 500 ms buffering.

The performance of the automatic parameter selection technique is close to the optimally selected static parameters. The automated method performs significantly

better for the 'hybrid' channel trace. This proves that the proposed method efficiently adapts to highly dynamic network conditions.

Table 3: Good channel - Comparison of video rate adaptation methods

Y-PSNR (dB)		Channel Trace: Good								
		HARBOUR			*CREW*			*RAVEN*		
Playout Buffer ⟶		0.1 s	0.2 s	0.5 s	0.1 s	0.2 s	0.5 s	0.1 s	0.2 s	0.5 s
Method	Bandwidth-Adaptive Non-Delay Constrained	35.16	35.16	35.16	39.28	39.28	39.28	42.23	42.23	42.23
	Single-Frame Delay-Constrained	35.20	35.27	35.27	39.34	39.37	39.37	42.30	42.33	42.33
	Static Time-Scale and R-D Optimized	35.26	35.27	35.27	39.36	39.37	39.37	42.32	42.33	42.33
	Automated Time-Scale and R-D Optimized	35.00	35.25	35.27	39.32	39.36	39.37	42.16	42.31	42.33

Table 4: Deteriorated channel - Comparison of video rate adaptation methods

Y-PSNR (dB)		Channel Trace: Deteriorated								
		HARBOUR			*CREW*			*RAVEN*		
Playout Buffer ⟶		0.1 s	0.2 s	0.5 s	0.1 s	0.2 s	0.5 s	0.1 s	0.2 s	0.5 s
Method	Bandwidth-Adaptive Non-Delay Constrained	32.80	32.81	32.81	37.81	37.81	37.81	40.33	40.33	40.33
	Single-Frame Delay-Constrained	31.03	31.70	32.16	37.35	37.61	37.44	39.52	39.89	39.94
	Static Time-Scale and R-D Optimized	34.23	34.56	34.48	38.45	38.61	38.55	41.33	41.63	41.58
	Automated Time-Scale and R-D Optimized	32.75	34.46	34.62	38.20	38.57	38.62	40.55	41.53	41.66

Figures 13, 14 and 15 plot the 10 second (600 frames) sample run of the streamed video quality in terms of PSNR when the initial playout buffer duration is set 100ms and 500 ms. The bandwidth variation during this sample experiment is plotted in Figure 12. The proposed technique prevents the large PSNR drops that can be caused by late packets when the playout buffer is limited to small values, e.g., 100 ms. It also consistently outperforms the bandwidth-adaptive non-delay constrained

Table 5: Bad channel - Comparison of video rate adaptation methods

Y-PSNR (dB)		Channel Trace: Bad								
		HARBOUR			*CREW*			*RAVEN*		
Playout Buffer ⟶		0.1 s	0.2 s	0.5 s	0.1 s	0.2 s	0.5 s	0.1 s	0.2 s	0.5 s
Method	Bandwidth-Adaptive Non-Delay Constrained	23.01	28.78	29.37	27.41	32.34	35.28	31.33	35.79	37.98
	Single-Frame Delay-Constrained	28.47	27.31	28.45	34.03	33.88	34.13	35.58	35.52	35.60
	Static Time-Scale and R-D Optimized	28.98	30.60	31.37	35.53	36.41	36.56	36.81	38.64	39.02
	Automated Time-Scale and R-D Optimized	28.97	30.42	31.49	35.55	36.38	36.66	37.40	38.44	38.92

Table 6: Hybrid channel trace - Comparison of video rate adaptation methods

Y-PSNR (dB)		Channel Trace: Hybrid								
		HARBOUR			*CREW*			*RAVEN*		
Playout Buffer ⟶		0.1 s	0.2 s	0.5 s	0.1 s	0.2 s	0.5 s	0.1 s	0.2 s	0.5 s
Method	Bandwidth-Adaptive Non-Delay Constrained	25.15	28.32	30.03	29.41	31.94	33.90	30.48	34.58	37.40
	Single-Frame Delay-Constrained	30.50	31.26	31.89	36.60	36.54	37.42	38.45	38.76	39.53
	Static Time-Scale and R-D Optimized	31.21	31.49	33.79	36.95	37.31	38.55	38.98	39.42	40.99
	Automated Time-Scale and R-D Optimized	31.98	33.40	33.79	37.45	38.02	37.94	39.71	40.73	40.88

rate adaptation method even if the initial buffer is large, e.g. 500 ms. The achieved quality is close to the PSNR of the input video stream when the buffering duration is 500 ms as shown in Figure 15. This demonstrates that our algorithm is able to sustain the quality expected from the high definition video even during the tough network conditions.

2.5 Conclusions

In this chapter of the thesis, we proposed and tested an application layer video rate adaptation method for improving the WLAN streaming quality. Video frame freezes

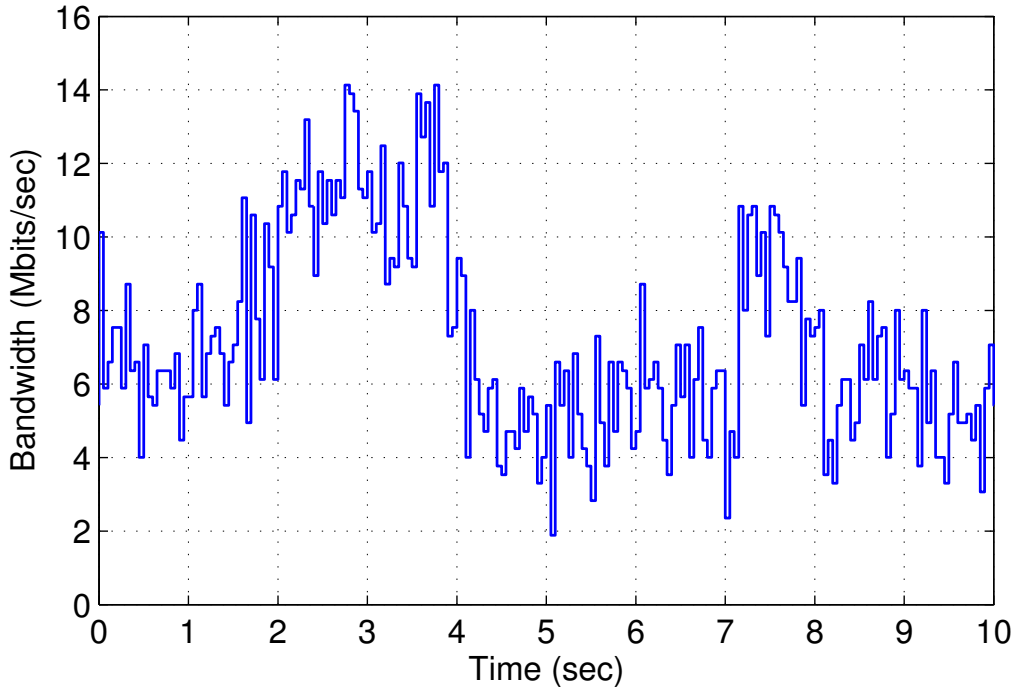

Figure 12: WLAN bandwidth - 'Bad' channel trace 10 second sample

and glitches caused by wireless bandwidth impairments are prevented by proactively adjusting the bitrate over various time scales. Our method also improves the transrater efficiency by a joint rate-distortion (R-D) optimization among a group of future frames. Up to 8.12 dB average PSNR improvement is achieved compared to a bandwidth-adaptive non-delay constrained rate adaptation method. We demonstrated that the proposed method enables the transmission of high definition video content over 802.11a/g WLANs.

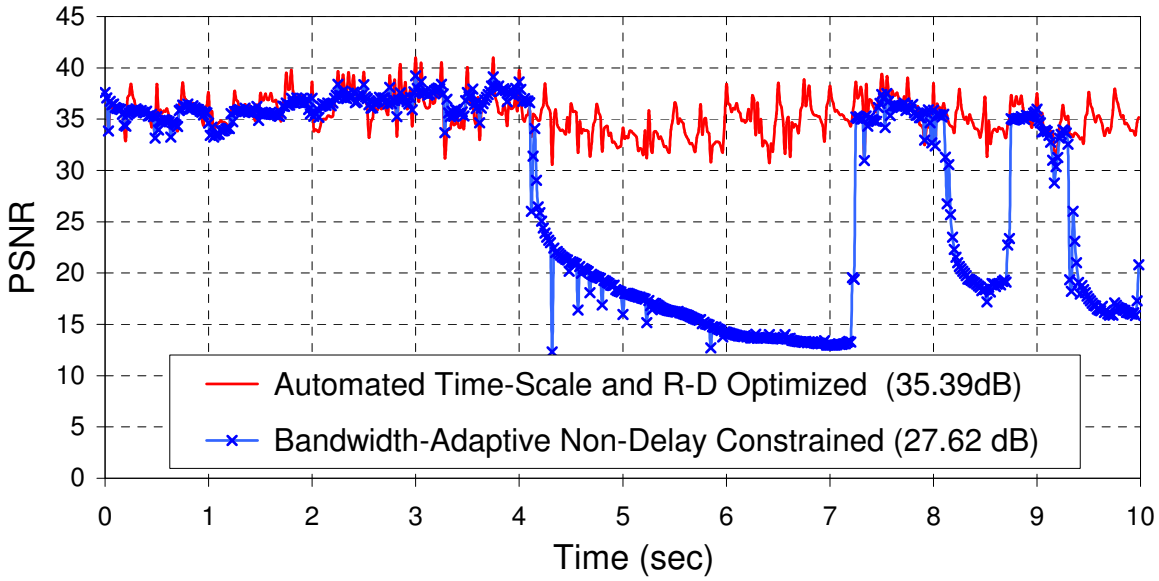

Figure 13: Comparison of video rate adaptation methods (Test sequence: CREW, Channel trace: 'Bad', Initial playout buffer: 100 ms)

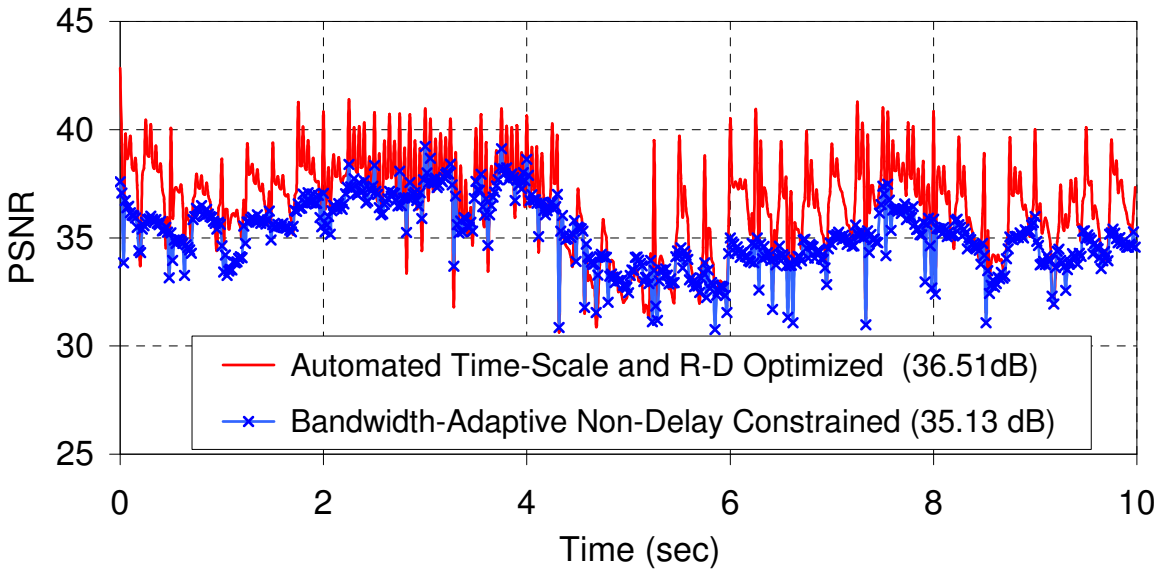

Figure 14: Comparison of video rate adaptation methods (Test sequence: CREW, Channel trace: 'Bad', Initial playout buffer: 500 ms)

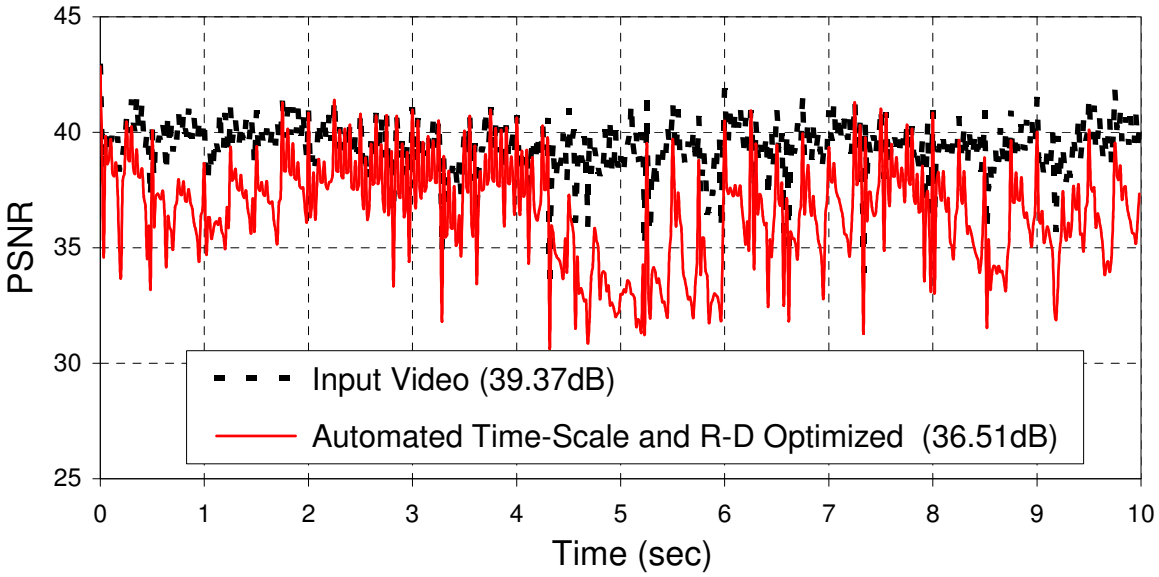

Figure 15: Comparison to the original video quality (Test sequence: CREW, Channel trace: 'Bad', Initial playout buffer: 500 ms)

CHAPTER III

RATE-ADAPTIVE WIRELESS TRANSMISSION OF VIDEO IN SCALABLE VIDEO CODING (SVC) FORMAT

3.1 Introduction

In Chapter 2 we explained our work on a rate-adaptive WLAN video streaming method, which used the MPEG-2 video coding standard. Streaming rate-adaptation for wireless transmission required transrating the MPEG-2 bitstream in real-time. The transrating requirement increases the cost of a home media gateway device since extra hardware and processing power is needed. This cost increases even more when more complicated closed-loop transrating methods are incorporated instead of simple open-loop techniques.

The requirement for transrating hardware/software can be eliminated if the video to be streamed is compressed in a scalable format. The scalability of the video refers to the flexibility in removing parts of the compressed video bitstream to provide compatibility with various devices/systems and adaptability to different network architectures. This flexibility most importantly enables playing the video on a large spectrum of devices ranging from mobile phones to wide screen TVs. In addition, scalability fits well into the heterogenous structure of today's wireless networks and Internet access technologies. For instance, video content servers can store a single version of each video, instead of separate copies coded at different bitrates, and offer service for broadband, dial-up or wireless clients.

Even tough MPEG-2 is widely used in DVDs and digital cable TV broadcasting, newly standardized H.264/AVC provides better compression efficiency and is replacing MPEG-2 in a variety of applications. The scalable video coding (SVC) standard

is developed as an annex of H.264/AVC. Although scalable video coding is not a new idea, which was previously used in the MPEG-2 and MPEG-4 standards, new coding methods used in SVC reduce the quality penalty that comes with scalability. Like its predecessors, SVC supports spatial, temporal and quality, i.e. SNR (signal-to-noise ratio), scalability dimensions.

The video delivery system targeted in this part of the thesis work is identical to the setup described in Chapter 2. The objective is to distribute high quality digital video content to display devices in a home environment over WLAN. Furthermore, we employ the wireless link quality probing tools previously developed. In Chapter 2 we described how the client devices measure the varying wireless bandwidth and provide real-time feedback to a media gateway device. The gateway device computes a transmission bitrate budget depending on the wireless bandwidth estimation. The differentiation of this work starts at this point. We will exploit the features of SVC to develop a more efficient and diverse set of robust WLAN streaming protocol level solutions.

The first novelty of this thesis work is on rate-distortion optimized allocation of the transmission bitrate budget. In chapter 2, we developed rate-distortion models for characterizing the importance of picture frames. These models were determined based on experiments that established the dependency on the resolution, bitrate and video sequence characteristics. The concept of quality layer (QL) identifiers has been introduced in the SVC standard to provide additional information that can be used for quality optimized extraction of low bitrate bitstreams. We will utilize QL information instead of R-D models for bit budget allocation.

The second differentiation is achieved by using temporal scalability. In Chapter 2 the frame rate of the video is retained regardless of the wireless bandwidth. When the link quality deteriorates significantly, reducing the quality without decreasing the frame rate may not be sufficient to sustain glitch-free video. The temporal scalability

feature of SVC coded video allows us to reduce the video rate even further to provide visually pleasing quality in adverse wireless conditions.

Another drawback of transrating-based rate adaptation is its multi-stage rate decision process. When the size of a video frame is decreased with transrating, it is not possible to revisit and refine this decision. The third major innovation of this work involves using the incremental data structure of SVC. Rate adaptation may aggressively reduce the bitrate to ensure that the client playout buffer never underflows when fast fluctuating bandwidth conditions are observed. These conservative decisions cannot be overturned. If transrating is used, however, SVC allows sending incremental refinement data, when the actual channel conditions are better than initially estimated. We proposed packet scheduling methods to exploit this feature.

In Section 3.2 we will explain the bitstream structure of the SVC coding standard. The developed rate adaptation and scheduling methods will be presented in Section 3.3. The performance of various streaming approaches will be compared in the experimental results section.

3.2 SVC Bitstream Structure

The bitrate of a full quality SVC video can be reduced in three dimensions. The first dimension is spatial scalability, where a video with lower resolution picture frames can be extracted. The temporal resolution, i.e. frame rate, of a scalable video may be reduced by simply discarding certain frames. The SVC standard enables temporal scalability by hierarchical B (bi-directionally predicted) and P (uni-directionally predicted) pictures. The third dimension is the quality or SNR scalability where the frame rate and resolution is preserved, however, the bitrate is controlled by adjusting the transform coefficient quantization levels. SVC provides both fine grain (FGS) and coarse grain (CGS) SNR scalability modes. FGS is achieved by encoding successive refinements of the transform coefficients, starting with the minimum quality

provided by H.264/AVC compatible intra / residual coding. The advantage of FGS over CGS is its ability to truncate progressive refinement (enhancement) layers at any particular rate point. Figure 16 illustrates the prediction, group of pictures (GOP) and layering structure of a sample bitstream with a GOP size equal to 4 and 2 SNR refinement layers. The first frame of the GOP (in encoding/decoding order) is a key picture. The key pictures are either intra-coded or inter-coded using previous (key) pictures as reference for motion compensated prediction. The remaining pictures of a GOP are hierarchically predicted B frames as illustrated in Figure 16. The number of temporal layers is determined by the size of the GOP. For the example depicted in Figure 16, there are three temporal layers. The lowest frame rate is achieved by only transmitting key frames.

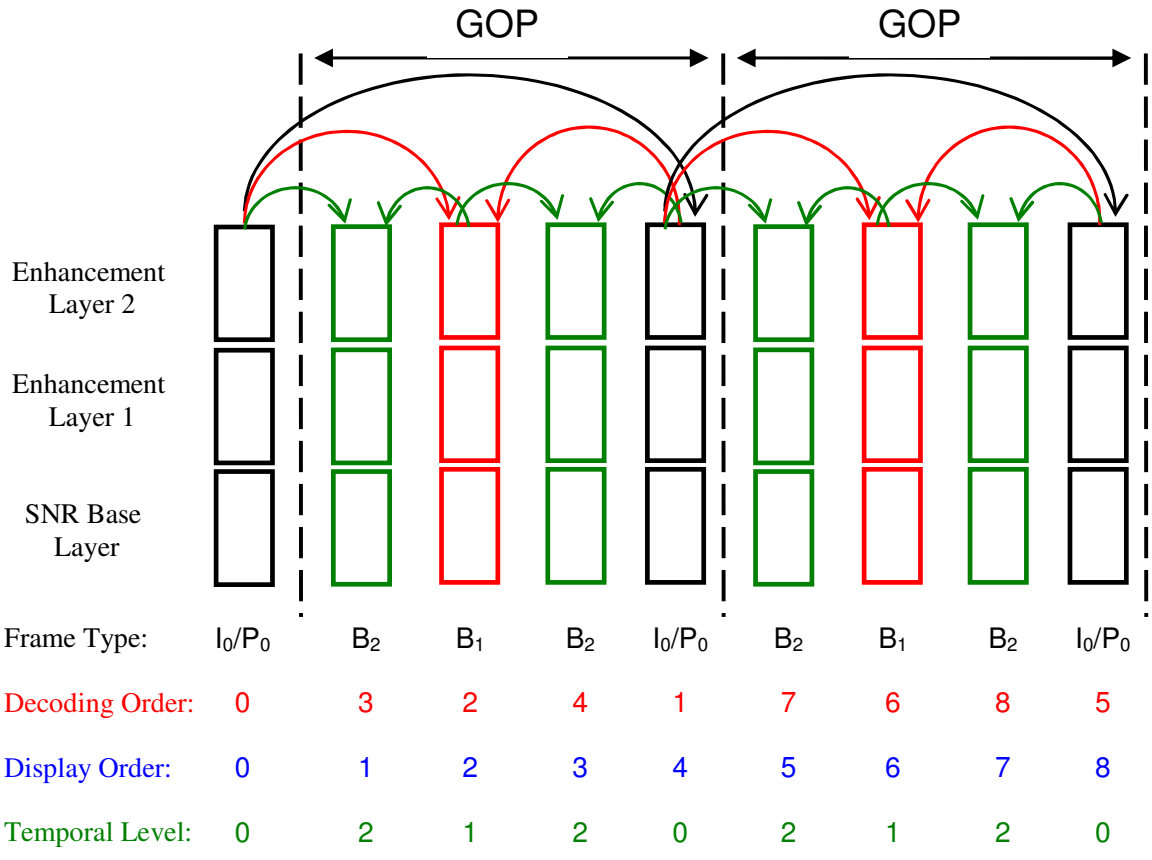

Figure 16: SVC bitstream structure

The SVC bitstream is composed of network abstraction layer (NAL) units similar to H.264/AVC. In H.264/AVC NAL units contain slice structures, which are independently decodable data units. Slices can be equivalent to or smaller than a coded picture. In SVC, the base layer data of a video frame is encapsulated into a NAL unit, and each quality enhancement layer of a video frame is encapsulated into different NAL units.

The concept of quality layer (QL) identifiers has been introduced in SVC in order to provide additional information that can be used for optimized adaptation of a scalable bit stream containing progressive refinement NAL units. The value of a QL identifier corresponds to the relative importance of a NAL unit in terms of the impact on the overall video distortion, if the NAL unit is not available for decoding, as well as the additional bitrate increase it creates. QL identifiers can be computed at the time of encoding by determining and ranking the rate-distortion slope (ratio of distortion and size) of the NAL units over the entire bitstream. The value of a QL identifier gets higher as the relative importance of the NAL unit increases.

According to the SVC standard draft and reference model, "Joint Scalable Video Model" [2], quality enhancement layer identifier values can be selected between 0 and 63 (inclusive). We denote the SNR enhancement layers of frame i by i_k, $k = 1, ..., L-1$. In this notation the total number of SNR layers, including the base layer, is represented with L. For the NAL unit i_0, i.e. SNR base layer of frame i, $QL(i_0)$ can take a value larger than 63. We set the QL identifier for every base layer NAL unit to QL_{max} value, which is 64. Figure 17 illustrates an example of QL identifiers for a video sequence consisting of a SNR base layer and two enhancement layers, i.e. $L = 3$.

Figure 18 demonstrates an example bitstream size distribution for three SNR layers. In Figure 18, $v(i_k)$ represents the initial size of the NAL unit i_k (in bits) before bit rate adaptation. The initial size of a picture frame can be computed

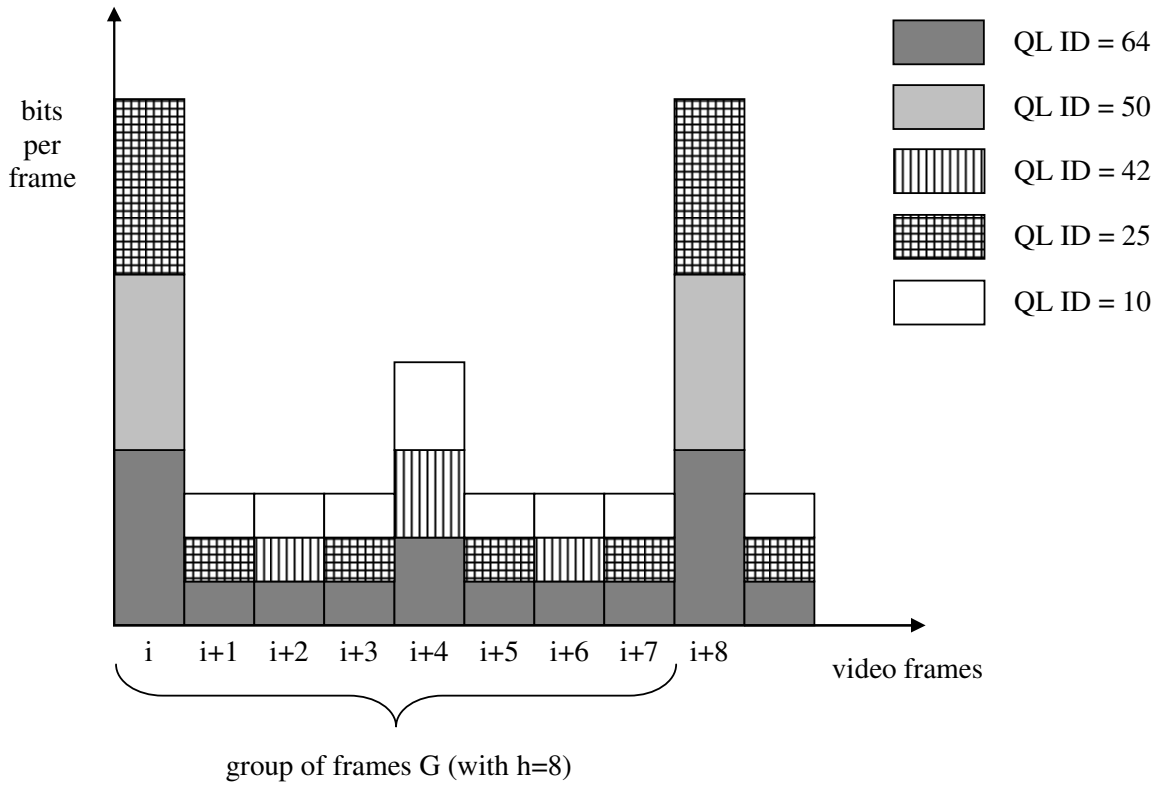

Figure 17: Quality layer (QL) identifiers of NAL units in a group of frames.

by summing the size of all frame i's SNR base and enhancement layer NAL units: $v(i) = \sum_k v(i_k)$. The initial average bitrate (bits/sec) of the video is denoted by V. $\tau(i)$ represents the temporal level of frame i. The temporal level of key frames are 0, as can be seen from Figure 16.

3.3 Rate-Adaptive SVC Streaming

In this thesis work we will develop real-time SVC stream bitrate adaptation techniques utilizing the SNR and temporal scalability features. SNR enhancement layers for each frame may be dropped (not transmitted) or may be truncated (partially transmitted) in order to reduce the bitrate. Furthermore, the temporal resolution of the video may be reduced up to a minimum level where only key frames are transmitted, by dropping non-key frames. The proposed rate adaptation methods are based on estimates of the

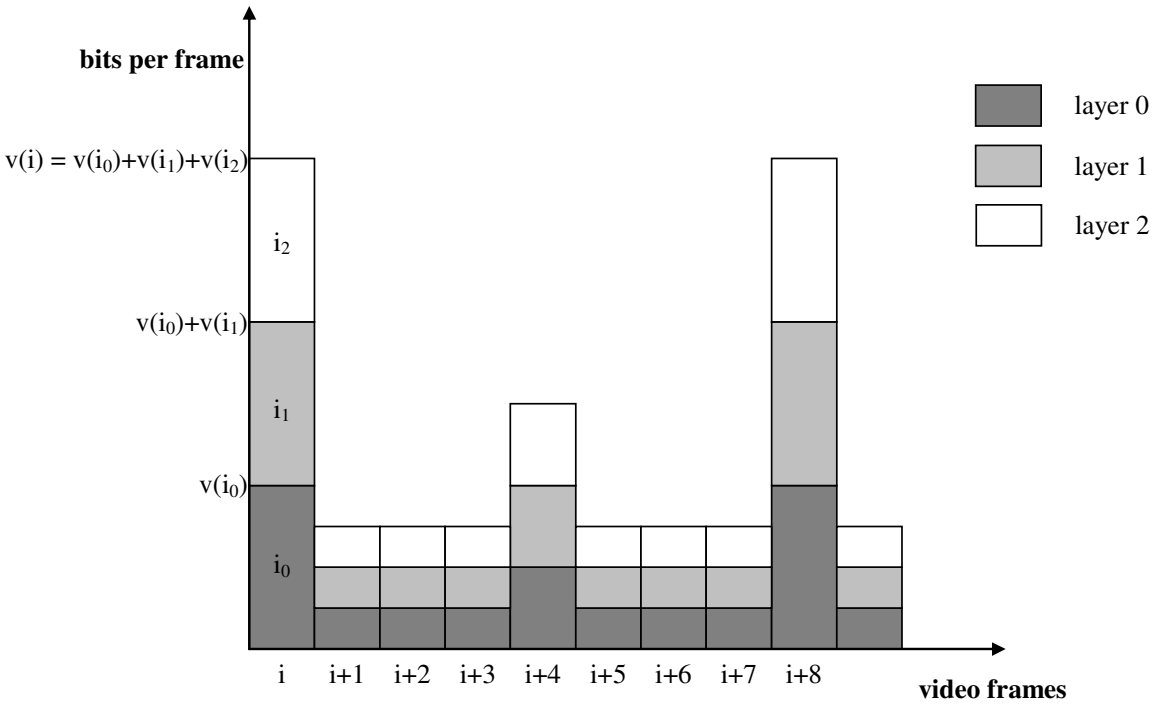

Figure 18: Example SVC bitstream NAL units and their bit sizes

channel bandwidth, $\hat{H}$, and channel backlog, $\hat{B}$. The channel bandwidth represents the maximum throughput and can be expressed in bits or bytes per second. The channel backlog represents the amount of data, expressed in bits or bytes, that is buffered somewhere in the channel, i.e. sent into the channel by the server, but not yet received by the client. For example, when streaming over a wireless LAN link, packets that are sent at the application or transport layer may still be held in a transmission buffer/queue at the sender MAC/PHY. The system may consider this data as yet to be transmitted. The channel bandwidth and channel backlog estimation techniques explained in Chapter 2 will be used in this study. The client sends feedback messages after receiving a burst of video packets. A burst is defined as a set of video packets that are sent at nearly the same time from the server application.

Real-time rate-adaptation is performed by considering the delay constraints of the video frames. Our objective is to deliver frames to the client prior to their decoding deadline, while minimizing the distortion due to rate adaptation. To this

end, we propose to utilize bitrate adjustment techniques based on multi-frame delay-constraints. In this approach, the current frame as well as multiple future frames in the transmission order are considered jointly for changing the streaming rate, using a pre-selected time scale. The first stage of this method involves the computation of a delay-constrained transmission bit-budget for these multiple frames. At the second stage, a transmission bit rate decision for the current frame is made. Quality layer (QL) information can be utilized for improved rate allocation. In the following, we first explain how the proposed SVC rate adaptation methods utilize SNR scalability and temporal scalability. Subsequently, an extension called "Delayed enhancement layer transmission" is described. This method improves the channel utilization in scenarios with a short initial playout buffer duration.

3.3.1 SVC Rate Adaptation based on Quality (SNR) Scalability

We first compute the delay-constrained bit-budget of the current, i, and $h-1$ future frames in the transmission order. These h frames are called a group and denoted as $G = \{i, i+1, ..., i+h-1\}$. The bit-budget for this group can be expressed as:

$$R_G = \hat{H}.[(h-1)\Delta T + F.\Delta T_E] - \hat{B} \tag{14}$$

In this expression R_G is the group bit-budget, ΔT is the inter frame interval, and ΔT_E is the initial playout buffering duration (delay tolerance). F is a parameter that determines the target size of the channel backlog at the end of transmitting the group of frames. The F parameter also determines the target fullness of the playout buffer at the client. Optimal values of the h and F parameters, which result in optimal quality, depend on delay tolerance, channel conditions and video source characteristics. For example, in a low delay scenario, the number of frames in a group h may be kept relatively low. These parameters can be extracted from look-up tables or automatically determined during the streaming session. The number of frames

in a group may be static (fixed) throughout a streaming session. Alternatively, the number of frames in a group may be varied dynamically, depending on channel and system conditions.

Next, we will investigate three possible cases. Cases will be determined by comparing the transmission bitrate budget to the size of the frame group size.

The first case occurs if the group bit budget is equal to or larger than the total size of all frames in the group, i.e. if $R_G \geq \sum_{j \in G} v(j)$. In this case all SNR layers of the current frame can be fully transmitted. Therefore, the rate of the current frame is not reduced, i.e.: $r(i_k) = v(i_k)$ for all $k < L$. $r(i_k)$ denotes the transmission size (in bits) of the NAL unit i_k after rate adaptation. $r(i) = \sum_k r(i_k)$ is the transmission size (in bits) of frame i after rate adaptation.

The second case occurs if the group bit budget is smaller than or equal to the total size of the SNR base layers of the frames in the group, i.e. if $R_G \leq \sum_{j \in G} v(j_0)$. The system will transmit only the base layer NAL unit of the current frame and discard all enhancement layer NAL units. Therefore: $r(i_0) = v(i_0)$ and $r(i_k) = 0$ for all $0 < k < L$.

The third case is when the group bit-budget is less than the total size of the frames in the group, but more than the total size of the SNR base layer data, i.e. if $\sum_{j \in G} v(j_0) < R_G < \sum_{j \in G} v(j)$. The system will transmit the base layer NAL unit of the current frame, and may transmit part of the enhancement layer data, and discard the remaining part. The bit rate of the stream should be reduced in the projected time scale that corresponds to the h frames in the group.

In this case, the transmission size of the current frame, $r(i)$, will be determined using the quality layer (QL) information, thereby distinguishing different video frames in terms of their effect on the overall video quality. The objective of this method is to minimize the distortion (i.e. maximize quality) by transmitting the most important NAL units in the group. Note again that more important NAL units have higher QL

identifier values.

For the third case, the transmission rate decisions for each enhancement layer NAL unit i_k $(0 < k < L)$ of the current frame i are determined as follows:

We define N as the set of NAL units in the group that are more important than the current NAL unit i_k. The elements of N satisfy the following condition:

$$j_s \in N \quad if \quad QL(j_s) > QL(i_k), \quad j \in G \quad and \quad 0 \leq s < L \tag{15}$$

The current NAL unit i_k is discarded entirely if the bit-budget is too small for the NAL units more important than i_k:

$$r(i_k) = 0, \quad if \quad R_G < \sum_{j_s \in N} v(j_s) \tag{16}$$

Otherwise, the transmission rate for the current NAL unit i_k is determined as follows:

Let us define P as the set of NAL units in the group G that have the same importance as the current NAL unit i_k. Thus:

$$j_s \in P \quad if \quad QL(j_s) = QL(i_k), \quad j \in G \quad and \quad 0 < s < L \tag{17}$$

The current NAL unit i_k is transmitted entirely if the bit-budget is large enough for all the NAL units belonging to the union of set N and set P.

$$r(i_k) = v(i_k), \quad if \quad R_G \geq \sum_{j_s \in (N \bigcup P)} v(j_s) \tag{18}$$

Otherwise the transmission rate assigned to the current NAL unit i_k can be calculated using any one of the following algorithm variants:

Approach 1: This approach treats all the NAL units in the set P equally and thus all the NAL units in P are truncated by the same ratio:

$$r(i_k) = v(i_k) \frac{R_G - \sum_{j_s \in N} v(j_s)}{\sum_{n_s \in P} v(n_s)} \tag{19}$$

Approach 2: This approach uses the remaining bits greedily by treating the current NAL unit i_k favorably compared to other NAL units belonging to the set P:

$$r(i_k) = \min(R_G - \sum_{j_s \in N} v(j_s), v(i_k)) \tag{20}$$

Note that truncation of this NAL unit is only allowed if the layer is coded in fine-grained scalable (FGS) manner. Otherwise, this NAL unit is discarded entirely.

3.3.2 SVC Rate Adaptation based on SNR and Temporal Scalability

If the rate reduction achieved by SNR scalability is not sufficient to prevent playout buffer underflows, futher rate adaptation can be performed by reducing the temporal resolution of the video. Temporal scalability is effectively achieved by discarding the SNR base layer data of non-key video frames. The proposed rate adaptation method based on temporal scalability again uses the time-scale concept. The goal of this method is to maximize the number of correctly decoded frames in the group while maintaining delay constraints.

The system decides whether the current frame is transmitted or dropped based on the importance of the frame in terms of its temporal level $\tau(i)$. Note that frames with a lower temporal level are more important than frames with a higher temporal level, since frames with a lower temporal level are used as references for prediction of frames with a higher temporal level. Figure 19 is an illustration of the temporal levels for 12 consecutive frames and GOP size of 8.

If the group bit-budget computed for SNR scalability is smaller than the total (bit) size of the SNR base layer NAL units of the frames in the group, i.e., $R_G < \sum_{j \in G} v(j_0)$, the temporal scalability feature is invoked. Temporal scalability may decide to discard the base layer data of certain frames, in addition to the enhancement layer data already discarded in the above condition.

In this method, the bit-budget is considered for a group of frames G^t, where the

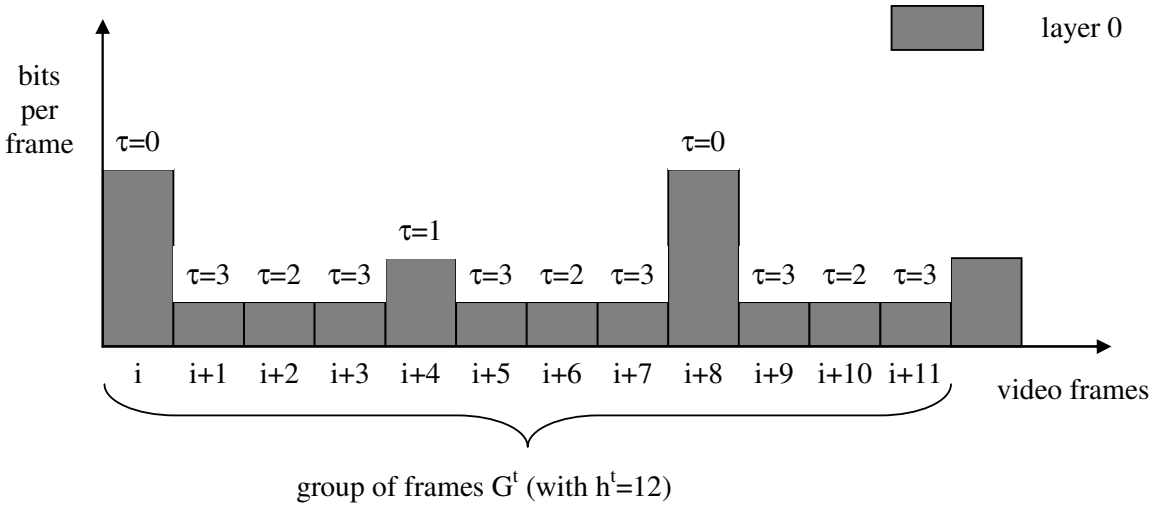

Figure 19: Temporal levels of base layer NAL units in the group of frames G^t

size of this group may be different than the size of the group G defined for SNR scalability. It is advantageous to use a larger group for the purpose of temporal scalability. Our experiments showed that choosing longer time scales for temporal scalability results in better performance. The new frame group is defined as: $G^t = \{i, i+1, ..., i+h^t-1\}$.

The transmission bit-budget is computed as follows:

$$R_G^t = \hat{H}.\left[(h^t-1)\Delta T + F^t.\Delta T_E\right] - \hat{B} \tag{21}$$

The base layer of the current frame (NAL unit i_0) is fully transmitted, if the bit-budget is large enough for this NAL unit as well as other NAL units in the group that are more important in terms of temporal level:

$$r(i_0) = v(i_0), \quad if \quad R_G^t \geq v(i_0) + \sum_{\tau(j)<\tau(i)} v(j_0) \quad for \quad j \in G^t \tag{22}$$

Otherwise, NAL unit i_0 is discarded, effectively dropping frame i entirely:

$$r(i_o) = 0, \quad if \quad R_G^t < v(i_0) + \sum_{\tau(j)<\tau(i)} v(j_0) \quad for \quad j \in G^t \tag{23}$$

3.3.3 Delayed Enhancement Layer Transmission

An additional method, named delayed enhancement layer transmission, is proposed as an extension of the previous two scalability methods. In the methods explained previously, NAL units have only one transmission opportunity at their generation time. These methods perform well when the initial buffering duration is relatively long by fully utilizing the channel when necessary. At short delay tolerances, the streaming system may not utilize the full channel bandwidth efficiently due to relatively large rate reductions. Large rate adjustments may occur when the h and F parameters are set to small values, to prevent buffer under-runs that may occur at fluctuating wireless bandwidth conditions.

The objective of this extension is to detect idle channel intervals (instances when the channel is not fully utilized), and subsequently transmit previously dropped or truncated NAL units whose decoding deadlines have not yet expired. Hence, better channel utilization is achieved by the delayed transmission of enhancement NAL units, i.e., at later transmission opportunities. This method also utilizes quality layer (QL) information for selecting the most important previously dropped or truncated NAL unit with a non-expired decoding deadline.

Idle channel intervals may be detected using feedback messages of the client, which are sent after a burst is received. The wireless channel will stay idle after the most recent burst transmission is acknowledged (i.e. the channel backlog is zero), and if there is still time left until the transmission time of the next frame.

First, the bit-budget for the idle interval is determined as follows:

$$R_I = \hat{H}.(t_{i+1} - c_{fb}) \tag{24}$$

In this expression t_{i+1} is the scheduled transmission time of the next frame $(i+1)$, and c_{fb} is the receive time of the latest feedback indicating that the channel backlog is empty. Delayed transmission can be disabled in cases where the idle interval is

shorter than a threshold (λ), i.e. $(t_{i+1} - c_{fb}) < \lambda$.

Next, the most important, not-expired and previously discarded/truncated NAL unit is determined. Note that SNR base layers are always more important than the enhancement layers. We define the set of NAL units eligible for delayed transmission, E, as:

$$j_k \in E, \quad if \quad t_j + \beta \cdot \Delta T_E \leq t_{i+1} \quad and \quad r(j_k) < v(j_k) \tag{25}$$

In this definition t_j is the initial transmission time of frame j. β is a constant safety factor, with $0 \leq \beta \leq 1$, that can be used to disable the delayed transmission of NAL units whose deadline is too close.

We then determine the most important NAL unit in E as the NAL unit with maximum QL identifier:

$$a_b = \arg\max_{j_k \in E} \left[QL(j_k) \right] \tag{26}$$

If multiple NAL units in E share the same maximum quality layer, the one with the minimum frame sequence number is selected. Next, the size of NAL unit a_b when performing delay transmission, $d(a_b)$, is calculated as follows. If $a_b$is an SNR base layer, i.e. $b = 0$, it will be transmitted entirely, since base layers cannot be truncated:

$$d(a_b) = v(a_b), \quad if \quad b = 0 \tag{27}$$

If a_b is an SNR enhancement layer, i.e. $b > 0$, the remaining bits/bytes of the NAL unit may be transmitted up to the calculated bit-budget:

$$d(a_b) = \min \left[R_I, v(a_b) - r(a_b) \right], \quad if \quad b > 0 \tag{28}$$

Note that truncation of this NAL unit is only allowed if the layer is coded in fine-grained scalable (FGS) manner.

The total transmission size of NAL unit a_b is updated after the delayed transmission:

$$r(a_b) = r(a_b) + d(a_b) \tag{29}$$

Finally, the client station may send a feedback message after receiving the delayed NAL unit packet burst. The above process may be repeated for the next most important NAL unit in E after updating the bit budget as follows:

$$R_I = R_I - d(a_b), \quad if \quad R_I > 0 \tag{30}$$

3.4 Experimental Results

The experimental environment considered in this study is very similar to the setup in Chapter 2. Sender and receiver stations are connected over an ad-hoc wireless link. Background data traffic was not allowed during the video streaming session. The AV streaming system is implemented in a simulation environment using MATLAB. In order to emulate realistic WLAN channel conditions, we collected packet traces using two laptop computers equipped with D-Link DWL-AG660 (802.11a/b/g) network interface cards. Application layer traces were collected by continuously transmitting 1500 byte IP packets from the server to the client, thus the channel never stayed idle. Using packet traces allowed us to compare different streaming methods fairly at identical conditions. The average bandwidth of the 802.11g channel trace used in the experiments is 4.27 Mbits/sec. For some experiments the usable portion of the channel bandwidth by the streaming application is artificially reduced by scaling the bandwidth.

Our simulator was also provided with video frame traces. HARBOUR and CREW test sequences at 704x576 pixels (4CIF) resolution, 3 Mbits/sec bitrate, and 60 Hz frame rate were used throughout the experiments. The number of SNR layers and the GOP size were set to 3 and 8, respectively. SVC reference software (JSVM 5.5) was used for encoding and decoding sequences. The bitrate composition of SNR and

temporal layers are summarized in Tables 7 and 8. The initial playout buffer duration was adjusted between 100 ms and 500 ms.

Table 7: Layer composition of HARBOUR test sequence (704x576 @ 60fps)

SNR Layer	Temporal Layer	Bitrate (Kbits/sec)
0	0	264
0	1	423
0	2	604
0	3	812
1	3	1708
2	3	3072

Table 8: Layer composition of CREW test sequence (704x576 @ 60fps)

SNR Layer	Temporal Layer	Bitrate (Kbits/sec)
0	0	248
0	1	430
0	2	658
0	3	1016
1	3	1723
2	3	3072

The WLAN streaming quality of five different methods, all using SVC, is compared. In the first method we find the maximum fixed rate that results in zero late frame percentage, by trial. The bitstream is extracted using quality layer information. This method cannot be used in a real streaming scenario; therefore it denotes an upper bound on the quality if the rate was fixed. It supports both SNR and temporal scalability. Streaming bitrates selected by this method are shown in Table 9.

The second method is called rate scaling. This method selects the rate on a per-frame basis using the bandwidth measurements. Only SNR scalability is supported (SNR base layers are always fully transmitted). The transmission size of the current frame is calculated as follows:

Table 9: Maximum fixed rate method - Video streaming rates

Video Bitrate (bits/sec)		Average WLAN Bandwidth (Mbits/sec)			
		4.27	3.20	2.13	1.06
CREW	100ms	1600000	1300000	658000	248000
	300 ms	3000000	2500000	1400000	658000
	500 ms	3072000	2600000	1500000	658000
HARBOUR	100ms	1600000	1100000	604000	265000
	300 ms	3072000	2500000	1600000	604000
	500 ms	3072000	2500000	1700000	604000

$$r(i) = min\left[1, F\frac{\hat{H}}{V}v(i)\right] \tag{31}$$

$\hat{H}$ is the bandwidth estimate, V is the initial video bitrate, and F is a constant safety parameter which is **0.5** (fixed in all experiments).

The third method is the SNR scalability method explained in Section 3.3.1. The time scale and maximum backlog target parameters shown in Table 10 are used in the experiments.

Table 10: h and F parameters for SNR scalability based rate adaptation

Initial Playout Buffering Duration	*h*	*F*
100 ms	2	0.1
300 ms	16	0.1
500 ms	24	0

The combined SNR and temporal scalability based method, explained in Section 3.3.2, uses the parameters in Table 11 when it switches to the temporal scalability mode.

The delayed enhancement layer transmission extension is the last method. $\lambda = 5ms$, and $\beta = 0.9$ are used in the simulations.

Figures 20, 21, 22, 23 show the performance comparison of the different methods at different WLAN banwidths. The delayed enhancement layer extension significantly

Table 11: h^t and F^t parameters for SNR + temporal scalability based rate adaptation

Initial Playout Buffering Duration	h^t	F^t
100 ms	8	0
300 ms	24	0
500 ms	48	0

improves the quality when the initial buffer duration is 100 ms. Up to 1.3 dB PSNR improvement is achieved compared to the rate scaling method. It also performs better than the maximum quality achieved by fixed rate selection in those conditions. The performance of all methods get close when the initial playout buffer is increased to 500 ms.

Temporal scalability is frequently used if the channel bandwidth is 1.06 Mbits/sec. Note that the quality of the methods that does not support temporal scalability is significantly reduced in low average channel bandwidth scenarios.

In Figures 24 and 25 we plot how the quality of the video streaming varies over time. We compare performance of the maximum fixed rate and delayed enhancement transmission methods. The average wireless bandwidth is selected as 3.2 Mbits/sec for these simulations. When the initial playout buffer is 100 ms, the maximum fixed rate streaming methods selects 1.1 Mbits/sec bitrate to ensure all SNR base layers are delivered on time. The video quality resulting from using this method is 28.32 dB, which is 2.69 dB lower than the performance of the proposed bandwidth adaptive SVC with delayed enhancement layer transmission. The improvement is 0.4 dB if 300 ms initial buffering is used.

3.5 Conclusions

In this chapter of thesis, we developed rate-adaptive WLAN video transmission techniques using the scalable video coding (SVC) extension of the H.264/AVC standard.

Fine-grain and temporal scalability dimensions of SVC are utilized in novel rate-adaptation methods. The proposed methods measure the channel bandwidth and delay statistics in real-time and adjust the transmission rate at selected time scales, considering the delay-constraints of the video. SVC quality layer (QL) identifiers are used for determining the importance of progressive refinement layers in terms of streaming distortion. Use of QL identifiers and flexibility in rate adaptation time scale selection maximizes the video quality while preventing playout buffer underflows. Rate adaptation is further improved with a delayed packet transmission scheme that solves the channel under utilization problem caused by bandwidth and delay estimation errors. Up to 1.3 dB average PSNR improvement is achieved compared to a non-delay constrained rate adaptation method, which also utilizes bandwidth measurements.

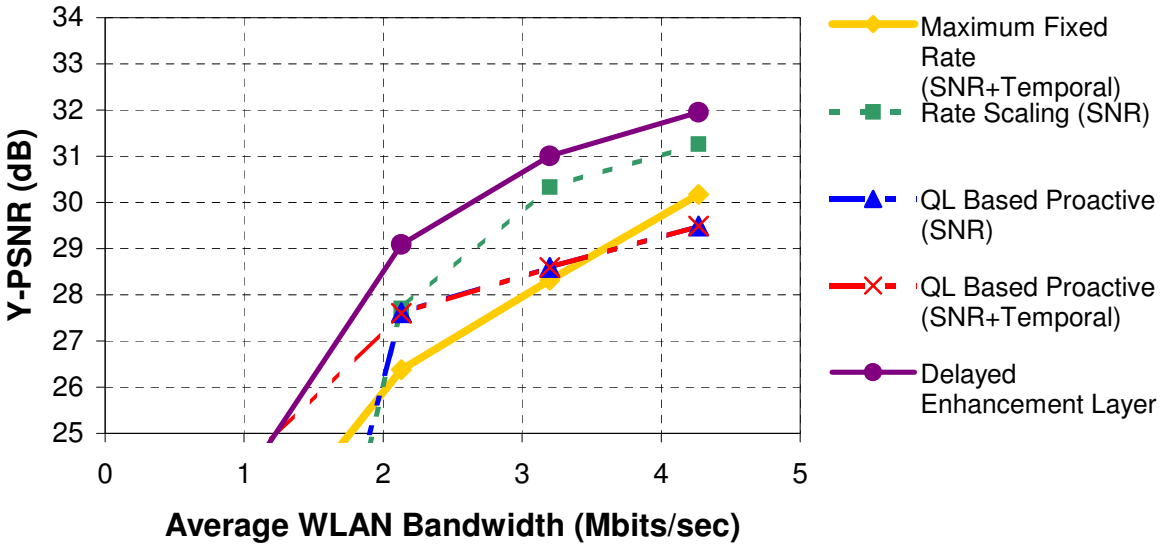

Figure 20: Comparison of SVC video rate adaptation methods - Test sequence: HARBOUR (3 Mbits/sec, 704x576 @ 60fps), Initial playout buffer: 100 ms

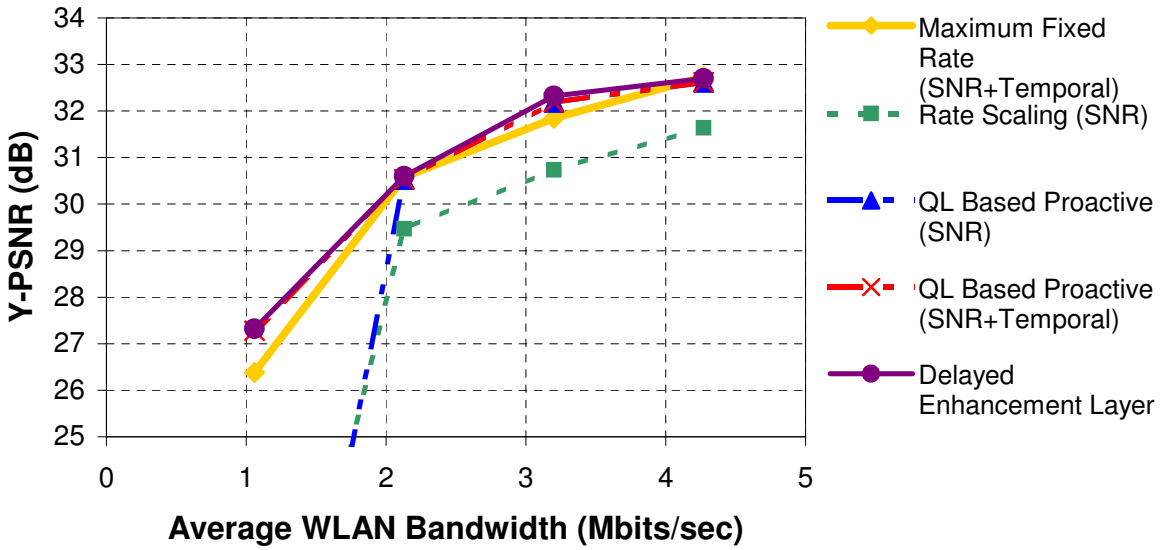

Figure 21: Comparison of SVC video rate adaptation methods - Test sequence: HARBOUR (3 Mbits/sec, 704x576 @ 60fps), Initial playout buffer: 500 ms

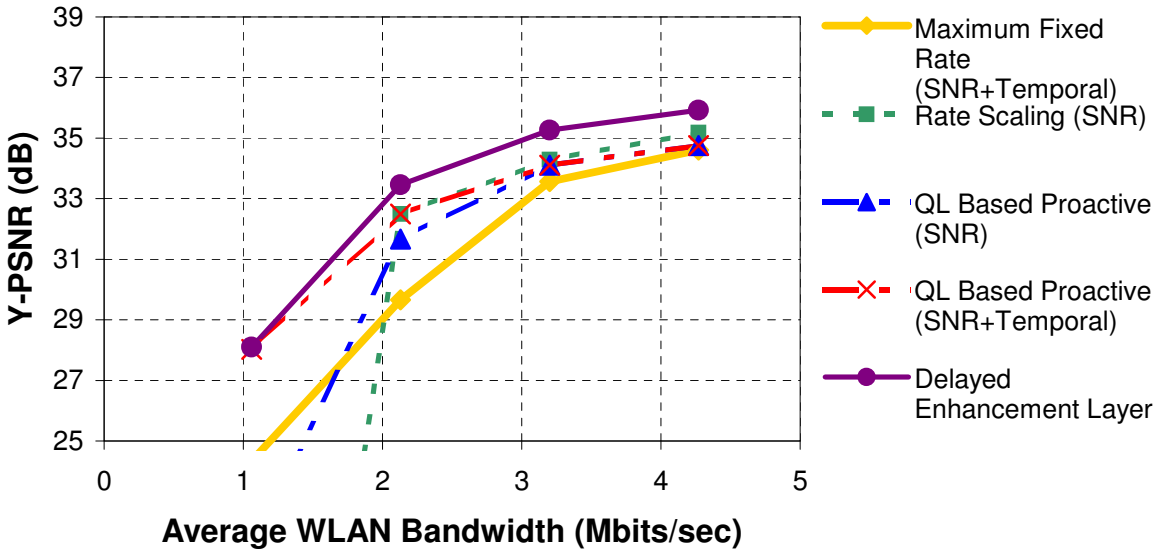

Figure 22: Comparison of SVC video rate adaptation methods - Test sequence: CREW (3 Mbits/sec, 704x576 @ 60fps), Initial playout buffer: 100 ms

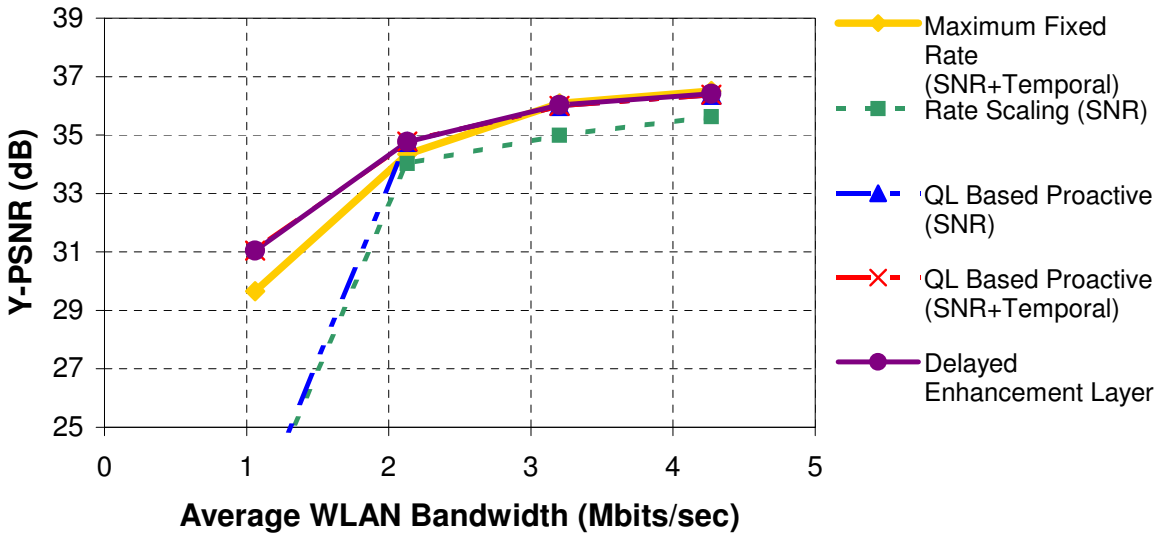

Figure 23: Comparison of SVC video rate adaptation methods - Test sequence: CREW (3 Mbits/sec, 704x576 @ 60fps), Initial playout buffer: 500 ms

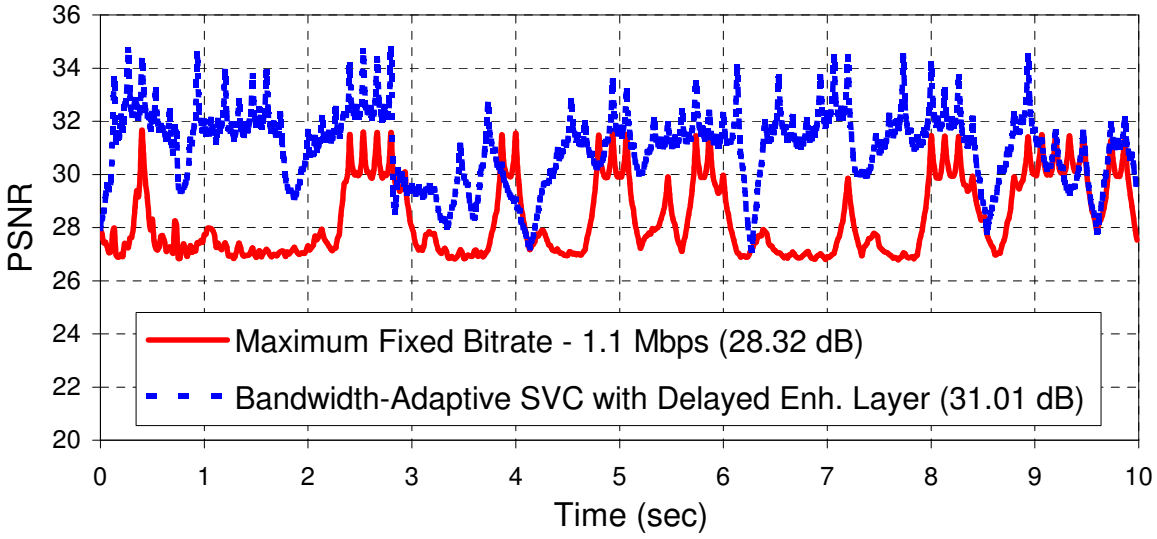

Figure 24: Comparison of SVC video rate adaptation methods, Test sequence: HARBOUR(3 Mbits/sec, 704x576 @ 60fps), Average WLAN bandwidth:3.20 Mbits/sec, Initial playout buffer: 100 ms

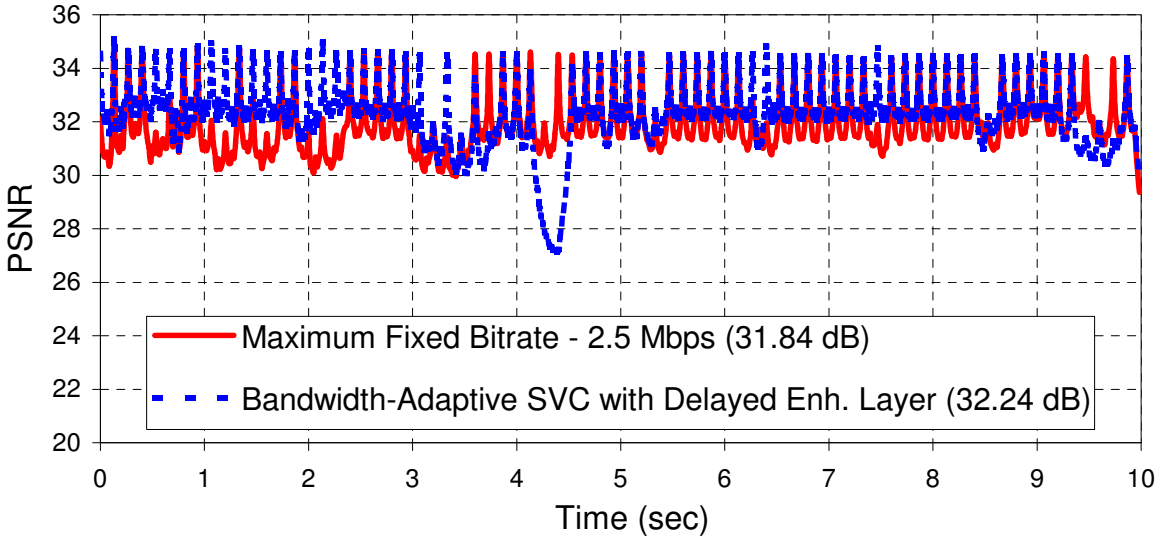

Figure 25: Comparison of SVC video rate adaptation methods, Test sequence: HARBOUR(3 Mbits/sec, 704x576 @ 60fps), Average WLAN bandwidth:3.20 Mbits/sec, Initial playout buffer: 300 ms

CHAPTER IV

FINITE-HORIZON FEC-RATE ADAPTATION FOR REALTIME WIRELESS MULTIMEDIA

4.1 Introduction

Transmission errors and packet losses are common problems of both wired and wireless networks. Protocols at various layers of the network protocol stack allow error detection and recovery. For instance, transmission errors can be corrected at the link layer on each hop of a network. Forward error correction (FEC) and automatic repeat request (ARQ) are most common schemes used at the link layer. End-to-end error control is performed at the transport layer. TCP employs retransmission techniques for the packets that the lower layers failed to deliver. Most of the widely used protocols do not differentiate among data and media packets. However, the concurrent transmission and display of media in real-time demands a fast error recovery. For example, the retransmission time-out duration in TCP is not acceptable for video-telephony and video-on-demand applications. Similarly, the code rate of the channel adaptive FEC may violate the timeliness requirements of the AV stream if it is excessively used.

Predictive coding in video prioritizes some data units over others. The importance level of a packetized media unit is determined by the amount of distortion caused in the multimedia presentation due to absence of that particular packet. For instance, in an MPEG video, the loss of a packet carrying slices of a reference frame, which is used to predict others, not only distorts that frame but also causes visual errors in its descendants. This behavior is called error propagation. Intuitively, using a higher FEC code rate for more important packets should provide a better video quality.

The FEC code rate selection for a packet effects the residual channel resources for subsequent packets because of the real-time constraints. If the sender chooses a high FEC rate for a packet, the transmission time of that packet increases, therefore, less time remains until the presentation deadline of subsequent packets. This may force the sender to use lower FEC rates than required (for these subsequent packets) to guarantee on-time delivery. These observations motivated us to develop an error-control method that jointly optimizes the FEC-code rates of current and subsequent packets considering their importance and deadlines. We will refer to the group of jointly optimized future packets as the optimization horizon. With the proper choice of the optimization horizon, we expect that the FEC performance does not deviate much from the optimality and processing requirement will be limited. The intra-coded synchronization frames and GOP structures in most video codecs bound the propagation range of the errors, hence motivate the limited optimization horizon.

A brief description of the wireless channel model and error rate estimation strategies are presented in Section 4.2. In Section 4.3, the optimization problem is formulated and an iterative solution algorithm is explained. Simulation results are presented in Section 4.4.

4.2 Wireless Channel Model

In this study, we assume a dedicated channel, which is the traffic mode for the real-time applications in the next generation cellular networks [4], [56]. The raw capacity of the channel before channel coding (number of assigned time slots) is fixed. In wireless mobile networks, packets are generally dropped or corrupted due to propagation errors. Mobility and fading cause the quality of a wireless channel to vary with time. The bit error rate (BER) at a given time is correlated to the previous channel conditions. Finite-State Markov Channel (FSMC) models are often used for the characterization of this kind of behavior [22]. In this study, we used FSMC models

to estimate the BER at a future time.

FSMC models can be constructed by partitioning the received signal SNR value into a finite set of ranges. Each range is mapped into a channel state. Several methodologies can be used for partitioning the SNR (*e.g.*, [65], [70]) and calculating the state transition probabilities. Each channel state corresponds to an average BER value. The client measures the signal SNR, determines the current state, and sends the state information to the sender periodically via the channel-state feedback messages. A channel BER estimation algorithm can be found in [17].

Let $S = \{s_0, s_1, \ldots, s_{M-1}\}$ be the set of M channel states and $q_{j,k}$ be the time independent state transition probability from state j to k, $0 \leq j, k \leq M-1$, given by

$$q_{j,k} = P\{S_{n+1} = s_k | S_n = s_j\}, \tag{32}$$

where S_n denotes the constant Markov process for time index $n = 0, 1, 2, \ldots$. The step time intervals (T) are determined in the SNR partitioning process. The channel noise characteristics and number of states are the factors that affect the length of the step interval. $\mathbf{Q}$ denotes the M-by-M transition matrix with its elements $q_{j,k}$.

$$\mathbf{p}^n(k) = P\{S_n = s_k\} \tag{33}$$

is the k^{th} element of the vector $\mathbf{p}^n$. $\mathbf{p}^n(k)$ is the probability of the channel being at the state s_k at the time index n. e_k is the BER associated with the state s_k.

In order to estimate the channel-state at a given time in the future, the latest channel state feedback message is used, assuming the Markovian property. If s_i is the channel state at time t_0 (time of the last feedback), then the state probability vector ($\mathbf{p}^0$) for the reference time t_0 is set to:

$$\mathbf{p}^0(i) = \begin{cases} 0, \; i \neq j \\ 1, \; i = j \end{cases} \quad \text{for } 0 \leq i, j \leq M-1 \tag{34}$$

The realization probability for each of the M states at a future time instant t_n, ($\mathbf{p}^n$) can be calculated as follows:

$$\begin{aligned} n &= \lfloor (t_n - t_0)/T \rfloor \\ \mathbf{p}^n &= \mathbf{p}^0 \times \mathbf{Q}^n \end{aligned} \tag{35}$$

The BER at n-steps in the future (ϵ^n) can be estimated as the state that has the largest realization probability.

$$\epsilon^n = e_k, \quad s.t. \quad k = \arg\max \mathbf{p}^n(k) \tag{36}$$

Note that we also assume that channel BER does not vary significantly during the transmission of a packet.

4.3 Problem Formalization and Solution Approach

The FEC rate decisions for the current and subsequent N packets in the transmission order will be jointly considered in the optimization process. This rate-adaptation will be re-executed periodically before the transmission of each packet. The output of the optimization algorithm will be the FEC redundancy rate for the current packet in the transmission order.

A packet's transmission time is determined by the selected FEC code-rate (r_l) for that packet. The transmission time increases with the amount of imposed redundancy as well as with the error-correction capability of the channel code. Figure 26 illustrates the scheduled transmission times of the current packet (packet i) and the subsequent packets for a possible code-rate decision vector $\mathbf{r} = (r_i, r_{i+1}, \ldots r_{i+N})$. Packets are transmitted continuously for the most efficient use of the dedicated channel. Next we will formulate a distortion function for each possible FEC decision, then we propose an algorithm for its optimization.

The gateway can choose from a finite set of channel code-rates $r_l \in \mathbf{C}$, $\mathbf{C} = \{c_o, c_1, \ldots, c_{k-1}\}$ for each packet. These code-rates can be generated easily by code

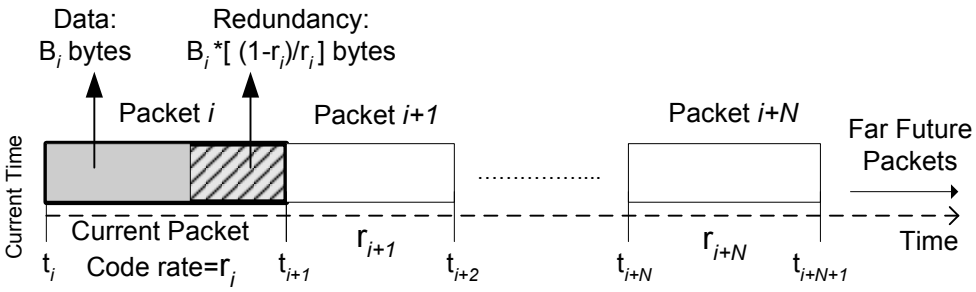

Figure 26: Current and subsequent N packets in the optimization range and their scheduled transmission times.

puncturing [13],[24]. The gateway also keeps track of the channel BER using the periodic channel quality indicators sent by the receiver. Utilizing a finite-state Markov channel (FSMC) model that characterizes the slow fading behavior [65], [21], [70], it then estimates the expected channel quality at a future time instant.

Let $P_{rec}(l) = f(\epsilon_l, r_l, B_l)$ denote the probability of recovering packet l from errors given the channel BER estimate (ϵ_l), selected code-rate (r_l) and the packet size (B_l). This probability lookup table can be determined by simulations or analytical computations. A packet should arrive before the presentation deadline to be useful. Increasing packet forward-trip times (FTT) may cause packets to be late. Therefore, by multiplying the probabilities of recovery and on-time conditions, we get the on-time successful delivery probability of a packet. That is, $P_{succ}(l) = P_{rec}(l) \times P\{t_{l+1} + FTT < t_{DTS,l}\}$, where t_{l+1} is the time instant that the transmission of packet l is completed, and $t_{DTS,l}$ is the decoding deadline for packet l. We model the error propagation phenomena and introduce a simple distortion measure for determining the effect of the packet losses on the video quality. A new probability, P_{dec}, is defined as the decodability probability of a frame that is encapsulated in packet l. We will use P_{succ} and dependency information in order to approximate P_{dec} as follows:

$$P_{dec}(l) = P_{succ}(l) \times \prod_{l' \in \mathcal{A}(l)} P_{succ}(l') \tag{37}$$

The product term in P_{dec} states that all of l's ancestors (reference frames, denoted by $\mathcal{A}(l)$) should be recovered successfully in order to be able to decode packet l.

The decodability probability of an intra-coded picture frame is equal to its on-time successful delivery probability since it does not depend on any other frame. We formulate our objective function (to be optimized) as:

$$\min_{\mathbf{r}} \left[D_0 - \sum_{l=i}^{i+N} \Delta d_l \times P_{dec}(l) \right], \tag{38}$$

where D_0 stands for the total distortion of the current and N subsequent frames (when they are lost), and Δd_l is the decrease in the distortion if packet l is displayed error free. The objective function computes the expected distortion of $N+1$ frames when the corresponding FEC code-rates are $\mathbf{r}$. Since the solution space size grows exponentially with N, we propose an iterative solution algorithm, which optimizes one variable (r_l) at a time until the objective function converges. After solving the optimization problem, the gateway forwards the current packet and the code-rate decision for it (r_i) to the base station. The optimization process is repeated before sending each packet, because of the variation in the channel state and content of the packets in the optimization range.

4.4 Simulations and Results

The efficacy of the proposed technique is evaluated through simulations with different channel characteristics and video bitrates. We compared performance of various other error-control strategies under the same conditions. In the simulations, a trace of the H.263+ encoded QCIF reference video sequence, FOREMAN, is used. To get more reliable results, the same sequence is concatenated back-to-back and a longer video (ten mins) is produced. The frame rate is set to 10 fps and the GOP size is selected as 10 frames. Each GOP consists of an I-frame and nine P-frames. The video frames are divided into 250 byte packets. Various video bitrates ranging from 40kbps to 90kbps are used in the simulations. The channel capacity is selected as 100kbps, which is a reasonable assumption when next generation systems are considered. The channel is modelled with a 11 state FSMC. The BERs corresponding to states and

state transition probabilities are taken from [70]. The BER varies in the range of 7.5×10^{-1} to 1×10^{-11} over time. RS codes are used for generating the redundancy. The number of possible FEC code rates (k) is 19 and the set is:
$C = \{defer, 1, 0.98, 0.96, 0.94, 0.92, 0.90, 0.86, 0.82, 0.77, 0.71, 0.67, 0.62, 0.59, 0.55, 0.52, 0.5, 0.33, 0.25\}$.

Performance of the developed optimized FEC rate adaptation technique is compared with three different alternative methods. In the first method, the FEC code rate is fixed for each packet and is set to the ratio of the video bitrate to the channel capacity. In the second method, the FEC rate varies according to the channel conditions. The code rate that reduces the packet corruption probability below a predefined threshold is selected. For instance, if the estimated BER is 10^{-3}, we select the code with the minimum redundancy level that delivers the packet successfully with a probability larger than 0.99. The third method considers the packet importance in addition to the channel conditions. The successful delivery probability threshold used in the previous method is adjusted according to packet importance. Stronger FEC protection is used for more important packets as a result of this modification.

Figure 27 depicts the comparison of these four methods at various bitrates. The qualities of the output videos are measured with the peak signal-to-noise ratio (PSNR) of the luminance (Y) channel. In these simulations, the playout delay is set to 0.5 seconds and the optimization horizon for the proposed algorithm is selected as 10.

For 40kbps video, all four methods perform (almost) equally well. At this rate 60% of the total bandwidth is used for channel coding, which corresponds to a high FEC code rate. The FEC code at this rate corrects almost all of the bit errors. As the video bitrate increases, the quality difference among the methods becomes more obvious. At 70kbps, the proposed method performs 0.7dB better than the source-channel adaptive method. The gain over fixed FEC method is around 1.95dB. From the plots, it can be seen that selecting a video rate that matches the channel capacity

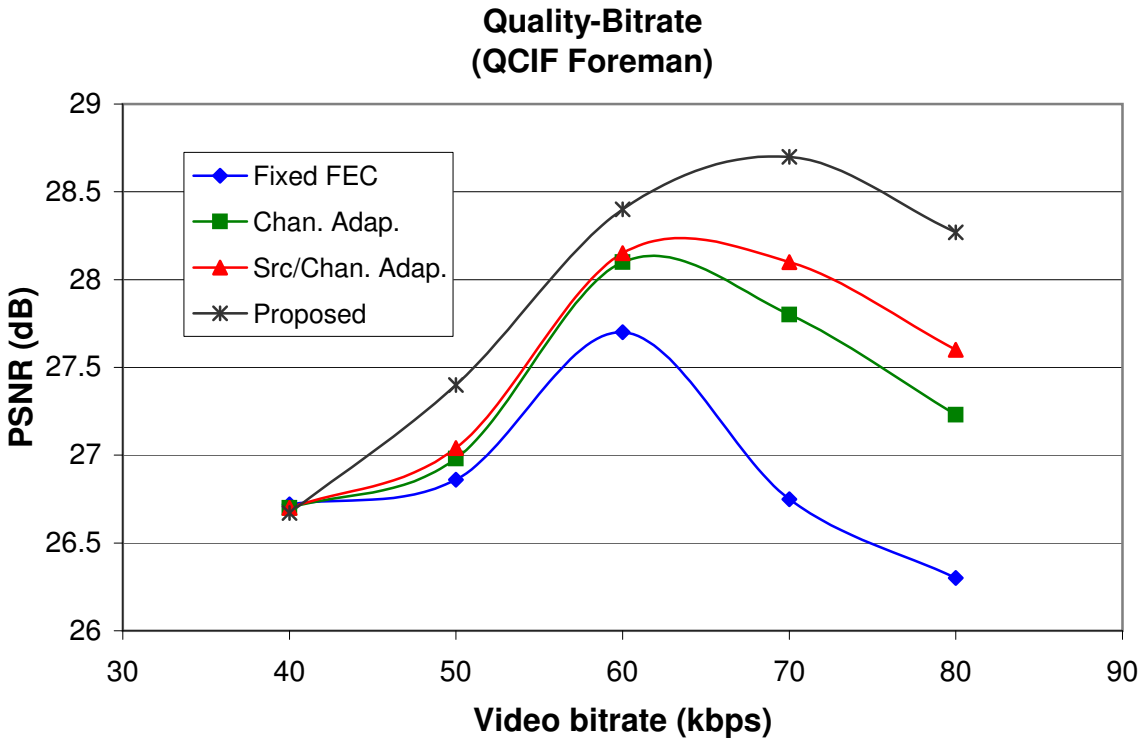

Figure 27: Quality comparison at various bitrates

is also another critical problem. The presentation quality suffers if a high bitrate video is streamed. Quality at 80kbps is less than 60kbps at the receiver, although 80kbps video was originally better.

If the error patterns in the previous simulations are examined, the channel adaptive and source-channel adaptive methods mainly suffer from the late packets. Delay tolerance is the main parameter that effects the percentage of late packets. We observed that if our technique is used, this percentage decreases. In the following simulations, our goal is to see the effect of the delay tolerance on the quality for all four methods. Results are presented in Figure 28.

Results show that the proposed technique achieves a higher gain at lower playout delays. The gain is about 1.8dB for 0.1 second playout delay over the source-channel adaptive method. As the delay tolerance increases beyond two seconds, the second,

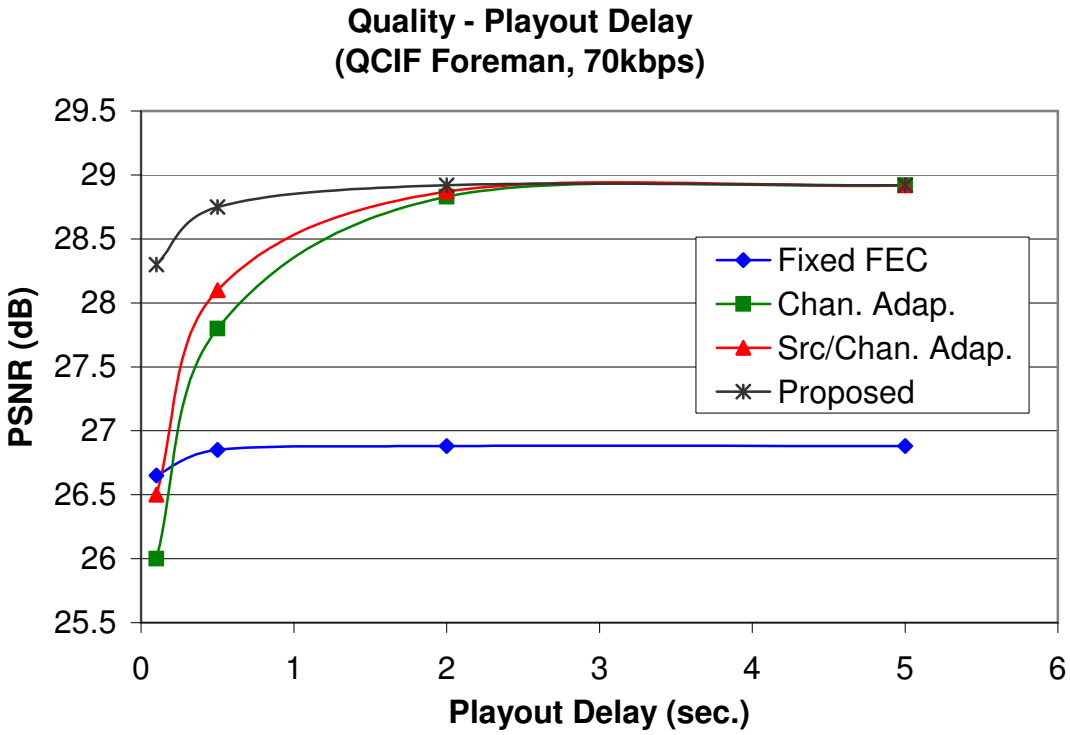

Figure 28: Quality comparison at various playout delays

third and proposed method perform approximately the same. At higher playout delays, the channel adaptive and source-channel adaptive methods do not suffer from late packets anymore. Based on the results in Figure 28, we conclude that our technique is particularly effective for interactive multimedia applications.

4.5 Conclusions

In chapter of the thesis, we proposed an optimization technique for FEC code rate adaption that improves the quality of multimedia streaming over wireless channels. We showed that the performance of channel adaptive FEC schemes can be improved when the media characteristics are considered. We proposed an optimization method that allocates the channel resources for packets, taking into account the BER estimate, packet importance, packets deadlines, and how this allocation would affect the quality in the future. In our method, the sender jointly optimizes the FEC code rates for the

current packet and future packets. It chooses the code rate for the current packet that maximizes the overall quality. A low complexity algorithm is presented for solving the formulated optimization problem. This algorithm makes the implementation of our technique feasible in real wireless communication systems. Simulations show that our technique increases the media quality significantly over non-optimized schemes for interactive and live applications.

CHAPTER V

PEER-ASSISTED VIDEO STREAMING WITH SUPPLY-DEMAND BASED CACHE OPTIMIZATION

5.1 Introduction

A peer-to-peer (P2P) network refers to an overlay structure where end users are able to exchange information among themselves without needing a central server. It is an alternative to the client-server paradigm commonly used in most web services. File sharing services over P2P networks have become one of the most dominating components of Internet traffic. Even though P2P has the bad reputation of facilitating illegal file downloading, it also has a potential for reducing the distribution cost of legitimate content. For instance, the file servers distributing software updates have to be over-provisioned in order to manage the flash crowds that occur following the time frame of the new releases. Over-provisioning significantly increases the hardware and bandwidth cost of content distribution systems. P2P collaboration can be useful in such scenarios by utilizing the unused upload bandwidth of the end users. In fact, the popular Bittorrent [1] P2P file sharing application is commonly used to distribute new Linux operating system releases. The use of P2P data distribution techniques for streaming live and on-demand video [38], [59], [34], [15], [68], [71], [18], [30],[42], [55], [32]. has become popular with the availability of high-speed network access technologies.

Video streaming is one of the most challenging services to offer because of the high and consistent bandwidth requirements of the digital video bitstreams. In a client-server based video service, a separate connection is opened for each client and data is unicasted to each of them. The number of simultaneous clients that can be served

by the system is limited by the server's disk performance and network bandwidth. Multicasting architectures offers a solution to this problem by enabling the replication of data at intermediate nodes of the network path. Therefore, the server does not have to send multiple copies of the same data to different users. Live video content such as TV programming can be distributed to many viewers over multicast trees without deploying high cost servers. However, multicasting-capable network routers are not widely deployed over the Internet and wireless cellular data networks. End-system multicasting is an application layer alternative to IP layer multicasting. Users in such a system act similar to the multicast routers and forward the data they have received to their peers, who are also viewing the same video. The asymmetry of downstream and upstream bandwidths in most wireless access technologies, such as 3G wireless, limits the maximum throughput that can be achieved by end-system multicasting. Furthermore, the users in a P2P application may leave the system any time they want, which reduces the reliability of the system. Therefore, pure peer-to-peer solutions are not very suitable for high quality video services. In this work, we focused on a hybrid architecture where end users assist central servers in video distribution.

Bittorrent [1] is currently the most popular peer-to-peer file sharing application. It also uses a hybrid structure, since a central tracker that keeps the record of users and their download status is adopted. A mesh network topology, where users locate and pull data from their peers, is formed. Bittorrent introduced new ideas such as segmentation of the file, downloading from random positions, rarest-segment first downloading and a tit-for-tat fairness mechanism. These ideas increased the availability of the files and total throughput by forcing users to share the data they are downloading and preventing free loading. These ideas on the other hand cannot be applied directly to video streaming. Video streaming requires segments of the video files to be downloaded in order. Furthermore, storing the whole video file may not be feasible due to storage and copyright concerns.

In this study we concentrate on on-demand video streaming services rather than live streams. In video on-demand (VoD) services, users start viewing the video at different time instants, and therefore, stream different segments of the file. It is hard to take advantage of multicasting based schemes in VoD because of this fact. Near on-demand solutions are proposed in the literature to reduce the load of the VoD servers. Most of these schemes involve periodic start times or user batching to utilize multicasting. Therefore, users wait for an initial time before starting to view the video. The waiting time is a negative factor that deteriorates the quality of the service perceived by the users, hence, it should be avoided as much as possible. Moreover, near on-demand solutions cannot provide full fast-forward or rewinding functionality.

5.2 System Model

The video-on-demand (VoD) service envisioned in this thesis study consists of control and video servers located in the wired part of the data network and wireless video clients (Figure 29). Clients are multimedia capable cellular phone and mobile device users subscribed to data services. In addition to being the receiver of the video, clients may also transmit video data from their caches to other peers. The control server is responsible for directing clients to video sources and coordinating the caching strategy of the clients. The video server(s) is the originator and main serving source of the video content. Video content may range from short clips like movie trailers, music videos, sport clips or advertisements to long clips like movies or TV shows. Each video clip is divided into equal size (in bytes) logical segments. The logical segmentation is a good way of quantizing the signaling time instants between client peers and servers.

When a client requests to view a clip, it sends a request message to the control server. The control server responds with a list of sources for the first segment of

the video file. This message exchange is depicted for the mobile client labeled as *Peer 1* in Figure 29. This list may contain the network addresses of video servers or other peers. If no peer is caching the requested segment or all caching peers are busy, the list only contains the address of the video server. The client probes the hosts in the list for resource availability. Based on the collected responses the client may decide to get the video file segment from one or multiple sources. For instance, *Peer 1* streams the first segment($S1$) from *Peer 2*. However, no peer is able to serve the fourth segment ($S4$), therefore, *Peer 4* gets this segment directly from the video server. The main bottleneck resource of a wireless video client is the upstream bandwidth because of the access technology asymmetry. If the video bitrate is higher than the available upstream bandwidth of a peer, that peer only serves a portion of the segment. In Figure 29, *Peer 3* streams $S3$ partially from *Peer 4* and the remaining bitrate is compensated by a video server. The signal exchange for locating the source is repeated before streaming each segment.

The VoD clients, i.e. peers, dedicate a limited portion of their memory or disk space for caching particular segment(s) of the video. Clients assist the main video servers in the content delivery process by uploading the cached file segments to other peers. As the efficiency of this peer collaboration improves, the load of the video servers will reduce, therefore more VoD clients can simultaneously stream video. Peer-to-peer collaboration is said to be maximized if the peers can fully use their upstream bandwidth during the time frame they are actively streaming video. The main objective this study is to design optimized caching strategies. The caching strategy decides which segment(s) of the file a peer caches and how long it caches.

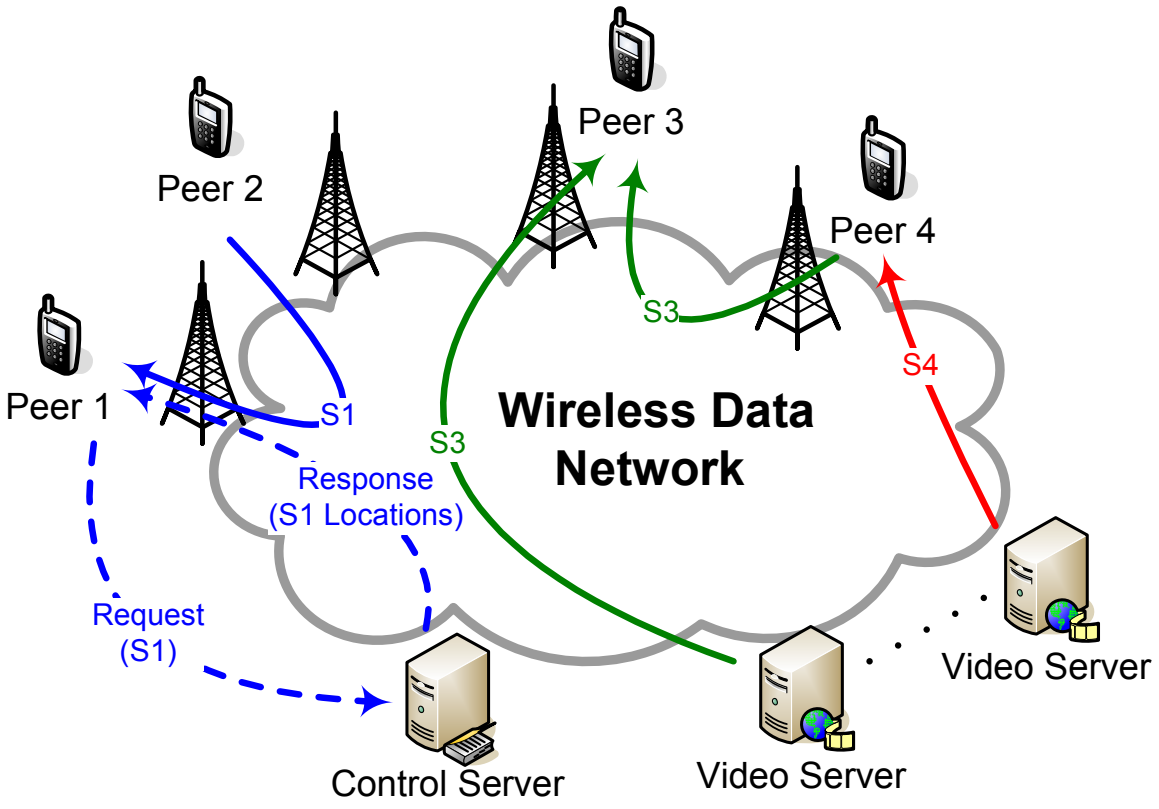

Figure 29: Peer-to-peer assisted VoD system architecture

5.3 Optimized Video Segment Caching for Peer-to-Peer VoD

5.3.1 Estimating Demand and Supply of File Segments

In a typical video-on-demand application scenario, users start viewing the clip from the beginning and progress through the end. After the clip has finished the user may leave the system or watch another video. However, this deterministic progress can be broken with random inputs from the user. Users may leave the system or switch to another clip before the video in progress has ended. Certain parts of the file can also be skipped. These stochastic actions are very hard or almost impossible to foresee. Therefore, in this study, for the sake of video progress estimation, we ignore random user inputs. However, there is still randomness in terms of stream start times (arrival times) of users. Figure 30 is an example illustration of the streaming start times of users and progress over the segments. A user continues to download the next file

segment immediately finishing one. The total download rate, or equally the streaming rate, is equal to the average bitrate of the video clip. This property is required to prevent over- or under-flowing of the client playout buffer. In Figure 30, the demand for each file segment is measured at time instant T. At time T, segments 1,2 and 3 are demanded by one user each. No user is streaming $S4$, therefore $D(S4, T)$ is equal to 0. The control server is able to measure the demands for segments of video clips at a given time since each user sends a request to it for each segment. The random nature of the user requests may result in an unbalanced demand for file segments. Figure 31 is an example snapshot demonstrating how many users are streaming a segment of the video clip at a given time instant.

The efficient planning of the caching strategies require the control server to estimate the demand for segments at a future time. Estimates are based on the latest measurements and streaming structure. We used discrete time-steps (epochs) to simplify the estimation process. The duration of one epoch is selected equal to the time it takes to fully download a file segment.

The measured demand for segment j at the current epoch, t_0 is denoted as $D(s_j, t_0)$. The control server targets to estimate $D(s_j, t)$ at the future epoch t. $D(s_j, t)$ can be decoupled into non-stochastic, $D_{ns}(s_j, t)$, and stochastic, $D_s(s_j, t)$, portions, i.e., $D(s_j, t) = D_{ns}(s_j, t) + D_s(s_j, t)$.

The deterministic portion is predicted assuming client streams the file to the end without any interruptions. Therefore, if a client is streaming segment s_{j-1} at epoch $t-1$ it will stream the subsequent segment, s_j in the next epoch, t. We express $D_{ns}(s_j, t)$ in recursive fashion based on this observation.

$$D_{ns}(s_j, t) = D_{ns}(s_{j-1}, t-1) \ ; \ \forall j \in \{2, 3, \ldots, M\}, \quad t > t_0 \tag{39}$$

The measured demand at time t_0 is denoted as:

$$D_{ns}(s_j, t_0) = \delta_j \ ; \ \forall j \in \{2, 3, \ldots, M\} \tag{40}$$

In those expressions, M is the number of file segments. The stochastic portion, $D_s(s_j, t)$ includes the clients joining the video stream after the current time, early terminated sessions and other user operations.

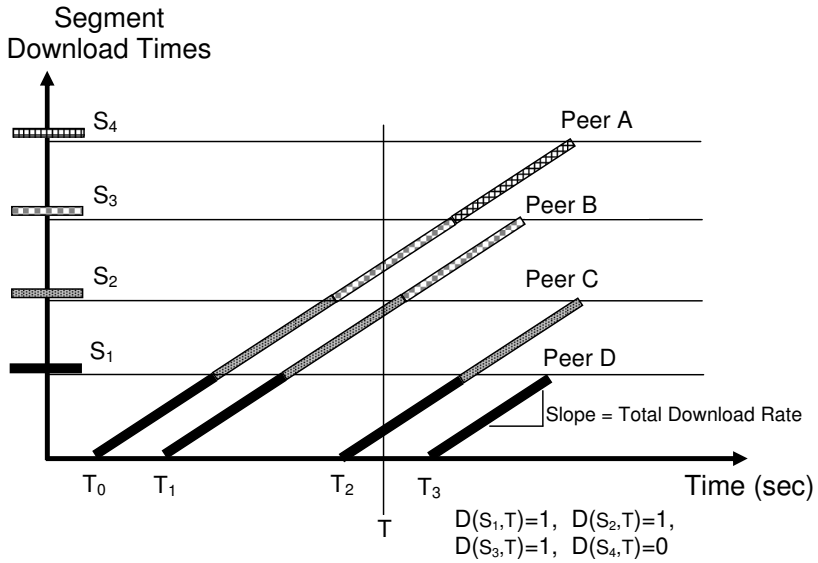

Figure 30: Evolution of the demand for the video file segments over time. Users arrive at random times and stream the video equal to the video bitrate. The overall demand estimate at time T is calculated

The other statistical parameter that will be defined in this chapter is the supply of a file segment. $S(s_j, t_0)$ denotes the measured total supply of segment s_j at current time t_0. Supply is equal to the sum of the available upload bandwidths of the peers caching segment s_j.

$$S(s_j, t) = \sum_{k \in \pi_{s_j}(t)} R_u^k \tag{41}$$

In Equation 41, $\pi_{s_j}(t_0)$ refers to the set of peers caching segment s_j and R_u^k is the available upstream bandwidth of a peer in that set. In this definition, we assumed a peer caches only one file segment at a time for the sake of simplicity. However, remember that the size of the cached segment can be adjusted by changing the number of segments in a video file. The supply also varies over time since new peers may join,

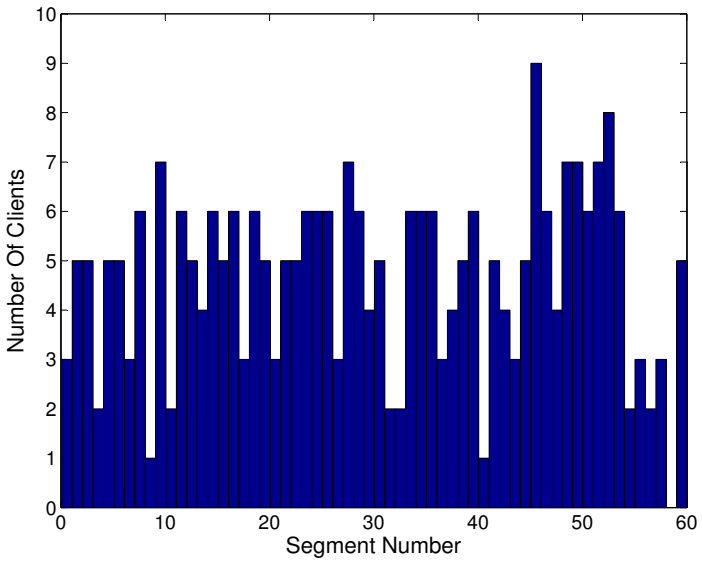

Figure 31: An example snapshot of demand for a segment of the file. Video length is 60 seconds, segment duration is 1 second and users are assumed to arrive with Poisson process of rate 5 users a second.

existing peers may leave or a peer may change the file segment it caches. We assume a client that finishes watching a video no longer serves other peers. It is not fair to utilize the resources of a client that is not receiving service. We also assumed that a client cannot start serving a file segment until the segment is fully downloaded.

Supply can be managed by the control server as opposed to the demand which is shaped by the clients. Therefore, efficient planning of the supply at future epochs $S(s_j, t)$ becomes the main tool of the optimized caching strategies. Maximum utilization of the user upstream bandwidth can be achieved if the supply can be adjusted to match the varying demand. In the following sections of this chapter we will define utility functions that quantify the degree of supply and demand alignment.

5.3.2 P2P Video Segment Caching Techniques

Caching techniques will be investigated in three categories. *Proactive Caching* constitutes the first category. Video segments that will be served to other peers are pre-fetched by the client within a data flow that is separate from the actual video

streaming traffic. In second category, which is named as *Reactive Caching*, users serve previously streamed file segments to their peers. *Hybrid Caching* is the third category that combines proactive and reactive caching approaches.

5.3.2.1 Proactive Caching

In this technique the VoD client downloads a video file segment from the video server or other peers for the sole purpose of serving it to other peers. Therefore, the caching data flow is independent of the actual video streaming traffic. Technically, the segments cached at a client can be updated regularly with this approach. However, if the pre-fetching traffic becomes dominant, less bandwidth will be available for the streaming hence the streaming quality will degrade. We will consider a scenario where a client only pre-fetches the segments to be cached when it initially joins the system. Furthermore, we assume pre-fetching takes place before the actual video streaming starts and that the download rate will be same as the streaming rate. Therefore, pre-fetching will not require extra bandwidth with the expense of an extra start-up delay that is equal to download time of cached segments. In this analysis we assume each client caches only one segment for the sake of simplicity. With this assumption the extra start-up delay becomes equal to the duration of a single file segment.

The control server has the responsibility of deciding which segment will be cached at which client. In the proactive caching strategy when the user demands a video file, the control server replies the sequence number of the video segment that the client should cache. Several methods on segment caching assignments can be employed at the control server. An intuitive and simple method is to assign segments in round-robin fashion. When a user is assigned to cache segment s_j, the next user that joins the streaming session will be told to cache segment $s_{mod(j+1,M)}$, and so on, given that M is the number of file segments in the video. This strategy targets the uniform distribution of the cached segments in the system. In another simple caching

strategy the VoD client may randomly pick the segments they will pre-fetch. An even distribution of cached segments can be achieved with this method in the steady state. This method also does not require the coordination of the control server.

Although the demand for file segments is expected to be uniformly distributed in the steady state, the snapshot of the demand may be unbalanced, as shown in Figure 31. Random streaming start times of the clients is the main reason for this behavior. Since the control server keeps track of the instantaneous segment demand snapshot, more intelligent caching assignment can be performed to handle the temporal demand variation. The objective of the proposed optimization is to shape the supply to match the demand. We define utility functions to measure the match level and determine which segments have a demand-supply mismatch.

For client k, who makes the initial streaming request before epoch t_0, the segment to be cached will be pre-fetched in the next epoch, t_0. Starting from epoch t_1 client k will start serving other peers. The utilization of client k's upstream bandwidth, if it serves segment s_j for the duration of h epochs, is computed as follows:

$$U(s_j, h) = \sum_{i=1}^{h} min \left\{ \left[\tilde{D}(s_j, t_i) - \tilde{S}(s_j, t_i) \right]^+, R_u^k \right\} \tag{42}$$

In Equation 42, $\tilde{D}$ and $\tilde{S}$ denote the estimated supply and demand for a segment over a given epoch. R_u^k is the available upstream bandwidth of k. $[x]^+$ is an operator that returns x if the value of x is larger than zero and returns 0, otherwise.

When the supply of s_j is higher than the demand for it, k will not be able to contribute to the peer collaboration, therefore the utilization will be 0. The maximum video uploading rate to other peers is limited by the supply deficit and the available upstream bandwidth of user k.

The control server's objective is to minimize the load of the central video servers by maximizing the usage of VoD client upstream bandwidths. Based on this goal, the newly joining user k will be told to cache the video file segment that results in

maximum utilization:

$$\underset{l \in 1,\ldots,M}{arg\,max}\ U(s_l, h) \tag{43}$$

The h parameter in the utility function (Equation 42) is the called the time horizon. For instance, if h is equal to the video length, i.e. M epochs, the utility is calculated for the entire time client k is streaming the video. On the other hand, only the next epoch is taken into consideration if h is set to one. Even though using a larger time horizon seems logical, it should be noted that future demand ($\tilde{D}$) and supply ($\tilde{S}$) estimates get less accurate as the time gap increases. We will analyze the effect of h parameter on the system performance in the results section.

The supply and demand estimates include deterministic and stochastic components as explained in Section 5.3.1. Demand estimation techniques that ignore the stochastic component will be referred as *blind estimation*. Stochastic demand, $D_s(s_j, t)$, involves newly joining clients, clients leaving early or forwarding /backwarding of the video by the users. Since the control server is able to measure the arrival rate of clients, it may use it to estimate future arrivals. Assuming that the contribution of early leaves and non-linear video play progress is minor compared to new arrivals, the expected value of the stochastic demand is computed as:

$$\begin{aligned} &D_s(s_j, t_i) = \lambda \quad ; \quad \forall j \leq i \\ &i \in 1, \ldots, h \\ &j \in 1, \ldots, M \end{aligned} \tag{44}$$

In Equation 44, λ is the average number of arrivals per epoch. In this stage we do not assume any specific arrival process.

Estimating the stochastic part of the supply is more challenging, since one should predict how the caching mechanism proceeds for all users in the future. However, when the new arrivals are totally ignored for supply estimation, a bias between demand and supply occurs. For the sake of unbiased utility function computation we

assume a client will continue to serve its cached segment even when it finishes streaming. This is equivalent to assuming a newly joining client will take over the cached segment of a departing client.

5.3.2.2 Reactive Caching

The reactive caching mechanism uses previously streamed file segments, hence does not involve pre-fetching. No extra delay or bandwidth is required by reactive caching in contrast to proactive caching.

At the beginning of the streaming session a newly joining client k starts streaming the first segment. Therefore, k does not cache and serve any file segments during the first epoch. After the first segment is fully downloaded, the client can serve it to other peers. When the second segment is downloaded, k should decide whether to store segment 2 or continue caching segment 1 (we assume each client caches only one segment). Therefore, after each epoch a client has two choices, keeping the already cached segment (s^*) or replacing it with the newly streamed segment (s_i). A possible caching decision tree is shown in Figure 32. Similar to the proactive caching method, the control server performs the centralized caching decisions for all VoD clients. The control server computes and signals the caching decisions to peers at each epoch.

In the reactive caching technique clients cannot cache the segments that they have not streamed yet, which is its main disadvantage compared to the proactive mechanism. Furthermore, if the clients choose not to cache a streamed segment, that data is lost and cannot be retrieved in the future.

We use a utility maximization approach to optimize the reactive caching decisions at each epoch for each user. The concept of time horizon is used to jointly optimize the current and future caching decisions.

Figure 32 illustrates possible caching patterns over the time horizon. Each pattern can be associated with a utility. Equation 45 represents the utility function of the

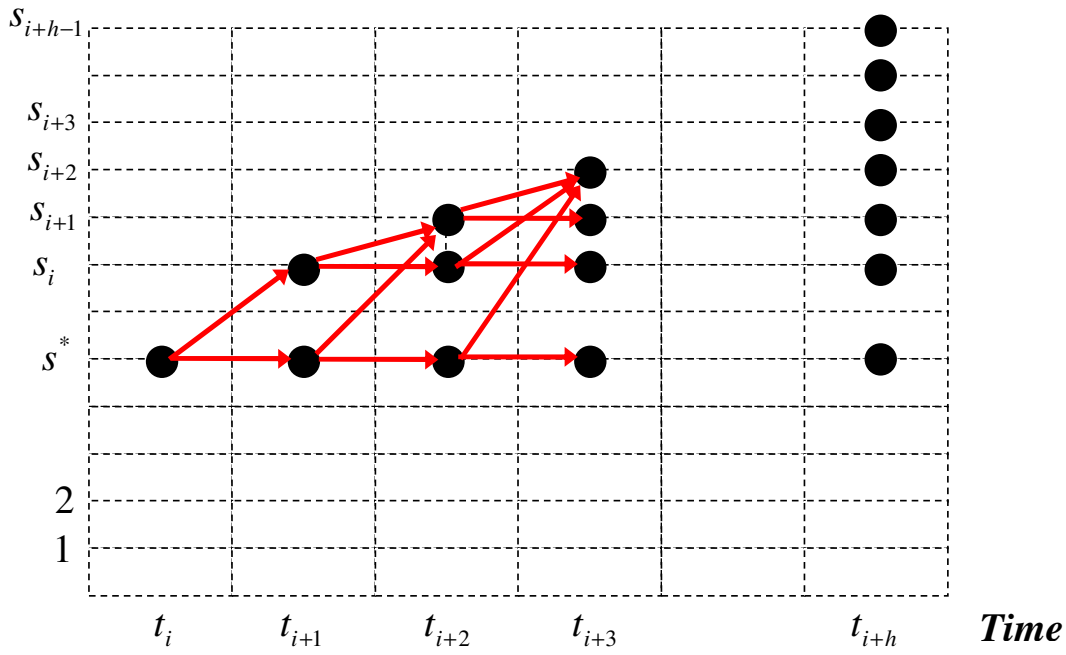

Figure 32: Caching decisions in reactive caching

caching decision sequence $c_J = \{c_{i+1}, \ldots, c_{i+h}\}$ of client k for the h future epochs.

$$U(c_J, h) = \sum_{j=i+1}^{i+h} \min\{\left[\tilde{D}(c_j, t_j) - \tilde{S}(c_j, t_j)\right]^+, R_u^k\} \tag{45}$$

Note that in Equation 45, c_j values may assume only two values. Referring to the example in Figure 32, c_{i+1} can be either s^* or s_i. The total number of different caching patterns over the time horizon h is 2^h. The set of caching patterns is represented by $\mathcal{P}_h$. The control server computes the optimal caching path using:

$$c_J^* = \underset{c_J \in P_h}{arg\,max}\, U(c_J, h) \tag{46}$$

Based on the result of this optimization, the optimal decision for the next epoch c_{i+1}^* is signalled to client k. The optimization is repeated for each epoch with the updated supply and demand estimates. Therefore future decisions, $\{c_{i+2}^*, \ldots, c_{i+h}^*\}$, are not executed.

Solving Equation 46 becomes expensive since the solution space size grows exponentially with h. Furthermore, the computation should be repeated for each user at

each epoch. We will utilize dynamic programming (DP) to solve this optimization problem using the fact that that the utility function is additive. The DP formulation is presented in Equation 47.

$$\max_{x^h \in \mathcal{X}^h} \left[q(x^h)\right] \quad for \quad \mathcal{X}^h = \{s*, s_i, s_{i+1}, \ldots, s_{i+h-1}\}$$

$$q(x^l) = \begin{cases} \max\limits_{y^{l-1} \in \mathcal{Y}(x^l)} \left[q(y^{l-1})\right] + \min\left\{\left[\tilde{D}(x^l, t_l) - \tilde{S}(x^l, t_l)\right]^+, R_u^k\right\}, & if \quad l > 1 \\ \min\left\{\left[\tilde{D}(x^l, t_l) - \tilde{S}(x^l, t_l)\right]^+, R_u^k\right\}, & if \quad l = 1 \end{cases}$$

$$\mathcal{Y}(x^l) = \begin{cases} \{s*, s_i, s_{i+1}, \ldots, s_{i+l-2}\}, & if \quad x^l = s_{i+l-1} \quad and \quad l > 1 \\ \{x^l\}, & if \quad x^l \neq s_{i+l-1} \quad and \quad l > 1 \\ \{s*\}, & if \quad l = 1 \end{cases} \tag{47}$$

$q(x^l)$ is the recursively defined utility function of caching state x^l at the l^{th} future epoch. States are shown with black circles in Figure 32. In Figure 32, directional arrows between two states symbolize possible transitions for two consecutive epochs. The set of previous caching states of x^l are denoted with $\mathcal{Y}(x^l)$. The solution of the dynamic program returns the optimal caching decision for the next epoch, which is either $s*$ or s_i.

The recursion defined in Equation 47 computes $q(.)$ for each possible state (nodes in the tree). Therefore, the complexity of the DP based algorithm is $O(h^2)$ and it can be computed as:

$$\sum_{i=0}^{h} i + 1 = (h+1) * (h+2)/2 \Rightarrow O(h^2) \tag{48}$$

The computational complexity of reactive caching can be reduced further at the cost of lower efficiency. The size of the solution space can be reduced to 2 by assuming that the segment that will be cached in the next epoch will be served over the entire time horizon. Therefore, the solution set $c_J = \{c_{i+1}, \ldots, c_{i+h}\}$ can either be

$\{s*, \ldots, s*\}$ or $\{s_i, \ldots, s_i\}$. This method will be called static optimal (SO).

5.3.2.3 Hybrid Caching

Hybrid caching combines proactive and reactive caching techniques. The control server decides which segment will be pre-fetched when a user starts streaming. This segment can be any segment of the video. Reactive caching decisions are performed at succeeding epochs.

5.4 Performance Analysis

The goal of peer-assisted video streaming is to cut the cost of deploying central video servers so that consumers can enjoy high quality media at a low price. To quantify this goal, we define our performance metric as the total upstream data rate of the central video servers.

We will compare the performance of proposed supply-demand optimized techniques to the simple techniques that are widely used in the literature. For this purpose we developed a video streaming simulation environment. The previously described peer-assisted video streaming system architecture is tested with simulating random video streaming requests.

The methods compared in the performance analysis are listed as follows:

- **NC + RR**: Selecting segments to be pre-fetched in round-robin fashion is a widely used method. We will label this method as no change / round-robin ($NC + RR$) since users do not change the pre-fetched segment until the streaming session ends.

- **NC + SO**: The supply-demand utility function optimized proactive caching technique will be labeled as no change / static optimal ($NC + SO$).

- **LP**: Caching the last playbacked (LP) file segment is a widely used reactive caching technique [18], [30]. This technique targets an even distribution of

cached segments in the system.

- **SO**: Proposed reactive caching technique with static optimal solution.
- **DP**: Proposed reactive caching technique with dynamic programming solution.
- **SO + RR**: Hybrid combination of static optimal reactive caching with round-robin pre-fetching.
- **SO + SO**: Hybrid combination of static optimal reactive caching with utility function optimized pre-fetching.

Other techniques used in the literature include caching rarest [6], most popular [46] or random [54] segments.

We will compare the performance of the listed caching approaches at various settings using the developed simulation environment. Several assumptions will be made to reduce the dimensionality of the simulations. We first assume that the users arrive, i.e. request to play a video, at random times that can be modeled as a Poisson process. The arrival rate per unit time (epoch) will be denoted by λ. We will assume all users have the same amount of cache size and upstream bandwidth resources. Users will cache only one file segment at a time. For the first set of experiments, each user's available upstream bandwidth allocated for P2P streaming collaboration is set equal to the video bitrate. Therefore, each user is able to serve a single other peer at the full video bitrate. For this first set of experiments we also assume users only leave the system when they complete streaming the whole video.

The video size is set to $M = 60$ segments. It takes a unit time for users to stream a single segment of the video. The normalized video server load depending on the arrival rate of the users is plotted in Figure 33. The normalized average video server load represents the ratio of total streaming traffic in the system that originates from the central server. The system performance improves as the average server

load decreases. The total streaming traffic in the system scales linearly with the number of users concurrently downloading the same video file. The average number of concurrent users can be calculated as $\lambda \times M$.

As shown in Figure 33, the normalized average server load reduces as users arrive more frequently, regardless of the caching method used. More frequent arrivals translate to more users concurrently viewing the same video file. When the number of concurrent users is small, the demand and supply of a file segment is less likely to match. Therefore, most of the peers are served by the central server rather than other peers.

The reactive caching mechanism in which users always cache and serve the last playbacked (LP) file segment performs significantly worse than other methods. The main drawback of this method is that the content of the cache is changed at each epoch. The proposed dynamic programming (DP) solution for supply-demand utility optimized reactive caching outperforms other techniques. For instance, when the arrival rate is 4 users per epoch (on average $4 \times 60 = 240$ users are simultaneously streaming) the central video server only supplies 10% of the total traffic, when DP is used. At the same arrival rate operating point, techniques other than LP, are clustered around 20% load. The proposed proactive caching technique ($NC + SO$) performs slightly better than the round-robin assignment of pre-fetching assignment ($NC + RR$). Hybrid proactive/reactive caching strategies ($SO + RR$ and $SO + SO$) perform better than proactive only ($NC + RR$ and $NC + SO$) and reactive only methods (SO and LP).

In this experiment, the optimization time horizon is selected as 4 epochs for the methods that use SO reactive caching. The dynamic programming solution used a time horizon that is equal to the number of epochs until the user completes video streaming.

In the next experiment we try to characterize the sensitivity of these methods to

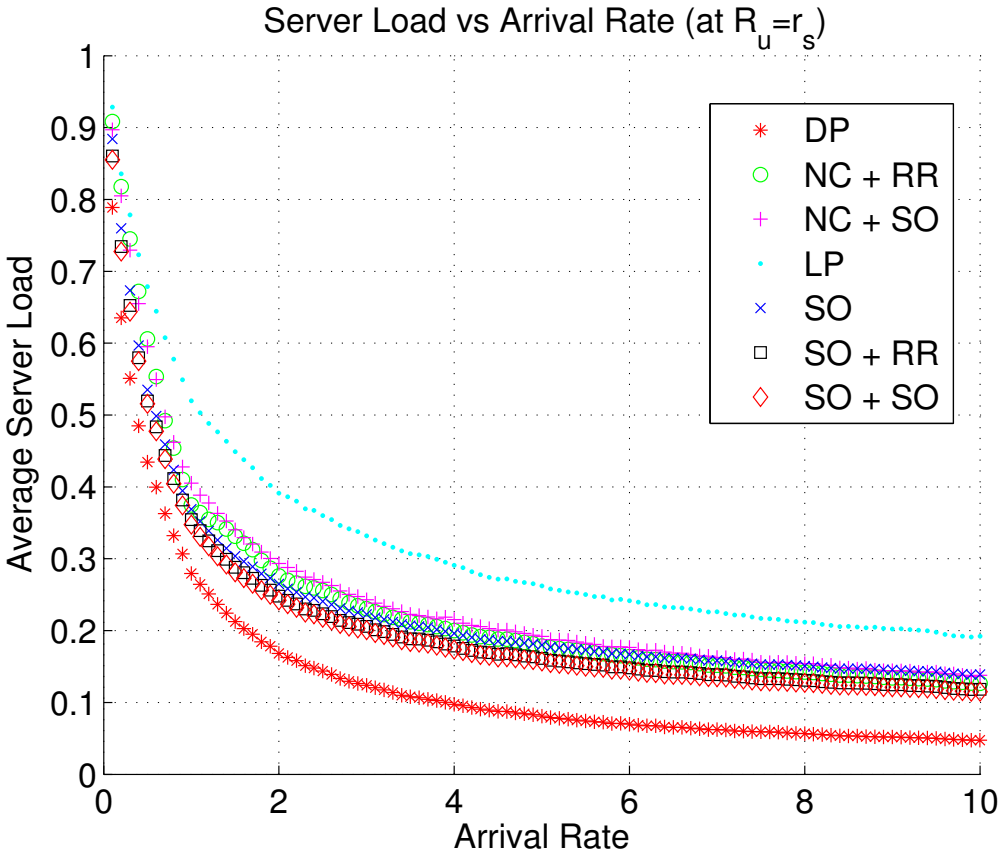

Figure 33: Normalized video server load depending on the user arrival rate (λ)

the selected optimization time horizon. We fix the user arrival rate at $\lambda = 1$. At this operating point, the variation of normalized average server load dependence on the time horizon is plotted in Figure 34. The *SO* technique achieves its best performance when the time horizon is set to $h = 4$. Note that the difference between this best performance and the highest server load point is only about 3%. When the time horizon is small, the utility function optimization based techniques try to fill near term supply-demand gaps. This approach may result in better performance since the predictability of future events decreases. For instance, the future supply for a video file segment is hard to estimate since it involves predicting the caching decisions of other peers in the system. The server load resulting from the *DP* decreases until

the optimization time horizon is 8. The performance variation after that point is marginal. Since the complexity of the DP based solution is $O(h^2)$ for each user, it is logical to select a smaller time horizon.

We also studied the effect of including stochastic demand terms, D_s, in addition to the estimated deterministic demand, D_{ns}. Stochastic terms were used to model demand due to future user arrivals. Blind estimation, i.e. not using stochastic terms, makes a very marginal difference in performance. The main reason for this behavior originates from the dynamics of reactive caching methods. Cached segments are frequently updated in reactive caching. Therefore, the demand for a candidate segment is almost deterministic for the relatively short time frame that the segment will be cached.

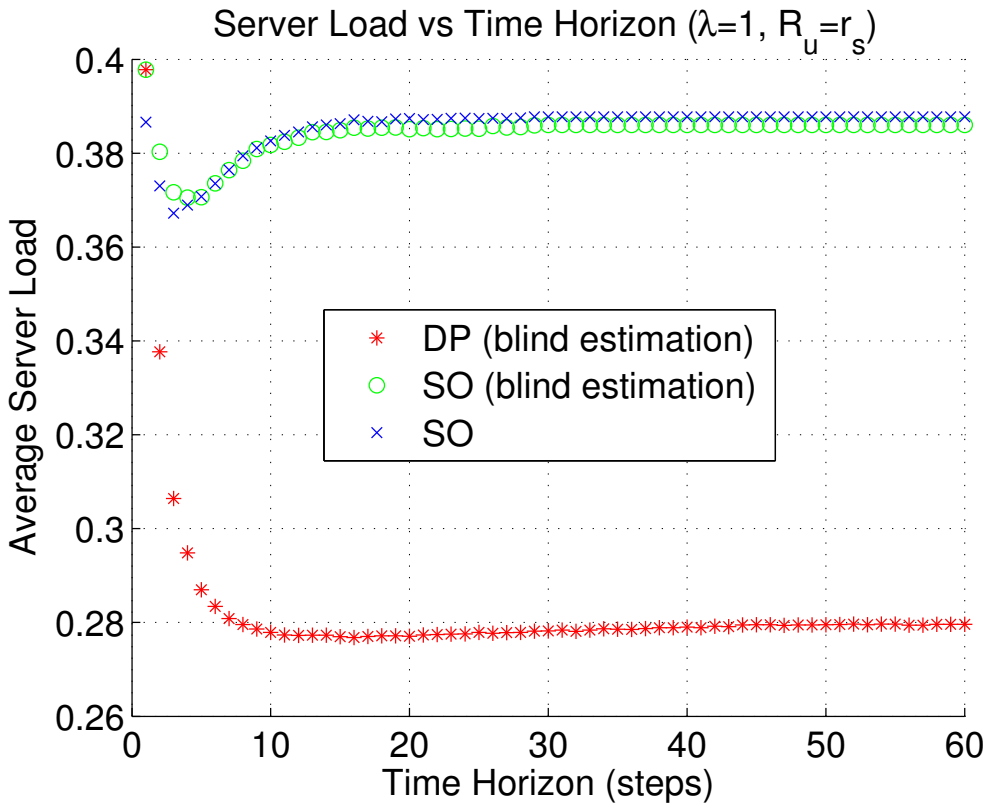

Figure 34: Effect of time horizon on normalized video server load

In the following experiment we relax the assumption that the available upload rate of peers is equivalent to the video streaming bitrate. The upstream rate to bitrate ratio will be changed in the 0-10 interval. The variation of average server load is investigated at a fixed user arrival rate ($\lambda = 1$). As expected, the performance of all methods improve as the upload rate gets larger (or video bitrate reduces). A user can concurrently serve 2 other peers when the upload rate is two times the streaming rate. Therefore, the central server has to serve fewer peers. More than one source should serve a client when the upload rate is less than the video streaming rate. The normalized server load reaches 1 for all techniques when the upload rate gets close to 0.

LP reactive caching again results in a higher server load compared to other techniques. The DP solution based reactive caching results in the lowest server load until the upload rate to streaming rate factor reaches 2, as demonstrated in Figure 35. After that point most of the methods perform almost equivalently. It is worth mentioning that round-robin (RR) pre-fetching becomes the best and static optimal (SO) reactive only caching performs about 8% less than that method. It can be concluded that proactive caching methods get more efficient than reactive only methods at high upload rates. This observation can be explained by the dynamics of caching methods. When the upload rate to streaming rate factor is high, the supply of a segment is generally larger than the demand for it. Therefore, the users do not have to update their caches frequently. Another advantage of proactive caching techniques is the flexibility in out-of-order pre-fetching of segments. Reactive caching methods can only cache previously played segments. In reactive mechanisms, the segments that are close to the end of the file are only cached by the peers that are about to leave the system. The last segment of the file cannot be served by peers in reactive only techniques because the users that played that segments immediately leave the system. Because of these limitations, reactive methods cannot create an evenly distributed segment

supply in contrast to round-robin (RR) based proactive caching.

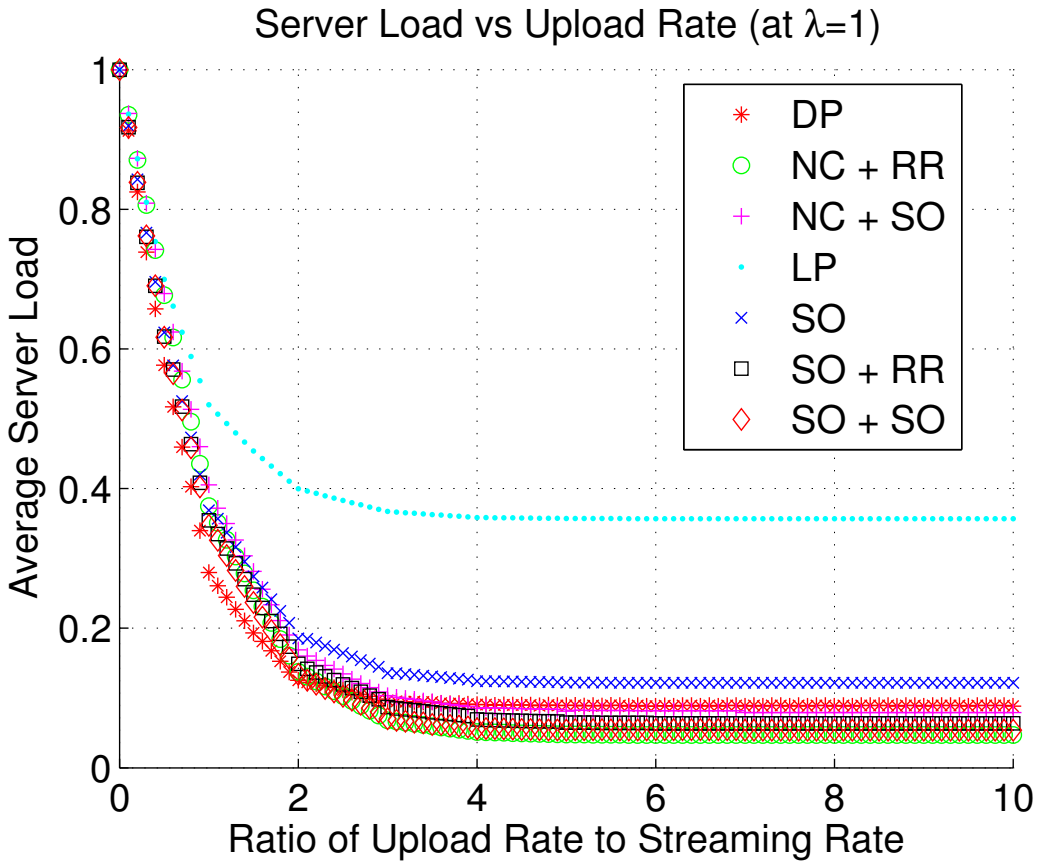

Figure 35: Effect of mobile client upload bandwidth on normalized server load

For the previous experiments we assumed the average arrival rate of the users over time is constant. However, in real video streaming systems it is common to observe flash crowds. Flash crowds occur when a large number of users request to play the same video file at nearby time instants. These irregularities may occur when the link for the video is shared in a mailing list or on a social networking site. We will try to model flash crowds as a superposition of two Poisson processes. The first component is the regular λ average arrival rate process used in previous experiments. The second component is an on-off modulated Poisson arrival process. Where a flash crowd occurs every T time units and lasts T_b time units. During the T_b interval users arrive with a

high rate, λ_{fc}, Poisson process. For the results demonstrated in Figure 36, T and T_b are set to 100 and 4, respectively.

Reactive caching methods show very good resiliency to flash crowds as shown in Figure 36. The average server load resulting from LP, DP, SO, $SO + RR$, and $SO + SO$ techniques decreases as the flash crowd rate increases. Please note that as the flash crowd rate increases, the overall arrival rate also increases. The reactive caching technique that caches the last playbacked LP segment greatly benefits from the flash crowds. It even outperforms the proactive only $NC + RR$ and $NC + SO$ methods after a point. When users arrive in flashes, they all stream the same or temporally close file segments. Therefore, the demand for a small subset of segments is large at a given time instant. The LP technique is good at capturing this behavior since the supply of this segment subset follows the demand with a single time unit lag. This demand variation behavior reduces the performance of pre-fetching techniques, as the flash crowds become more significant. For instance, the round-robin technique targets an evenly distributed segment supply, however, the flash crowds cause an uneven demand distribution.

$NC + SO$ proactive caching relies on estimating future segment demands. With the presence of flash crowds, the estimates get less accurate and $NC + SO$ gives the worst performance. The DP solution does a good job of adjusting the segment supply in the system and results in smallest server load.

In our final experiment we investigate how premature streaming session terminations affect the performance of various caching techniques. Early departures are modeled using a Poisson process. Figure 37 shows that the premature terminations increase the normalized server load for all methods. The user arrival rate (λ) is set to 1 and the upload to streaming rate ratio is selected as 1 for this experiment. The performance of $NC + SO$ proactive caching method exceeds $NC + RR$ when the rate of early departures reaches 0.3. This indicates that the supply-demand distribution

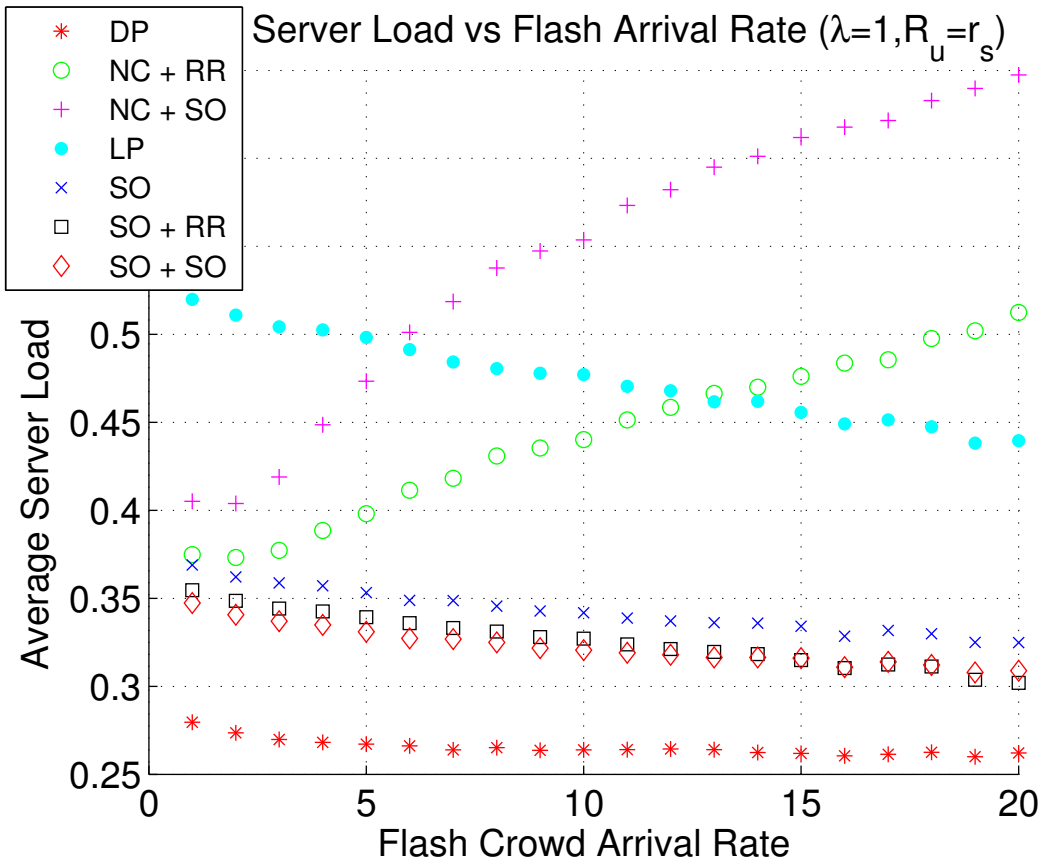

Figure 36: Effect of flash crowds on normalized server load

irregularities caused by early terminations are better handled with the utility function optimization based segment pre-fetching.

Dynamic programming based reactive cache scheduling using a supply-demand utility function is demonstrated to perform significantly better than other alternatives. *DP* achieves the lowest server load at the expense of increased computational complexity.

One aspect that is not considered in the simulations is the signaling overhead associated with the peer locating and caching mechanisms. Each user queries the download sources of a file segment at each epoch. This extra traffic overhead exists for all caching techniques. The size and frequency of signaling messages are small

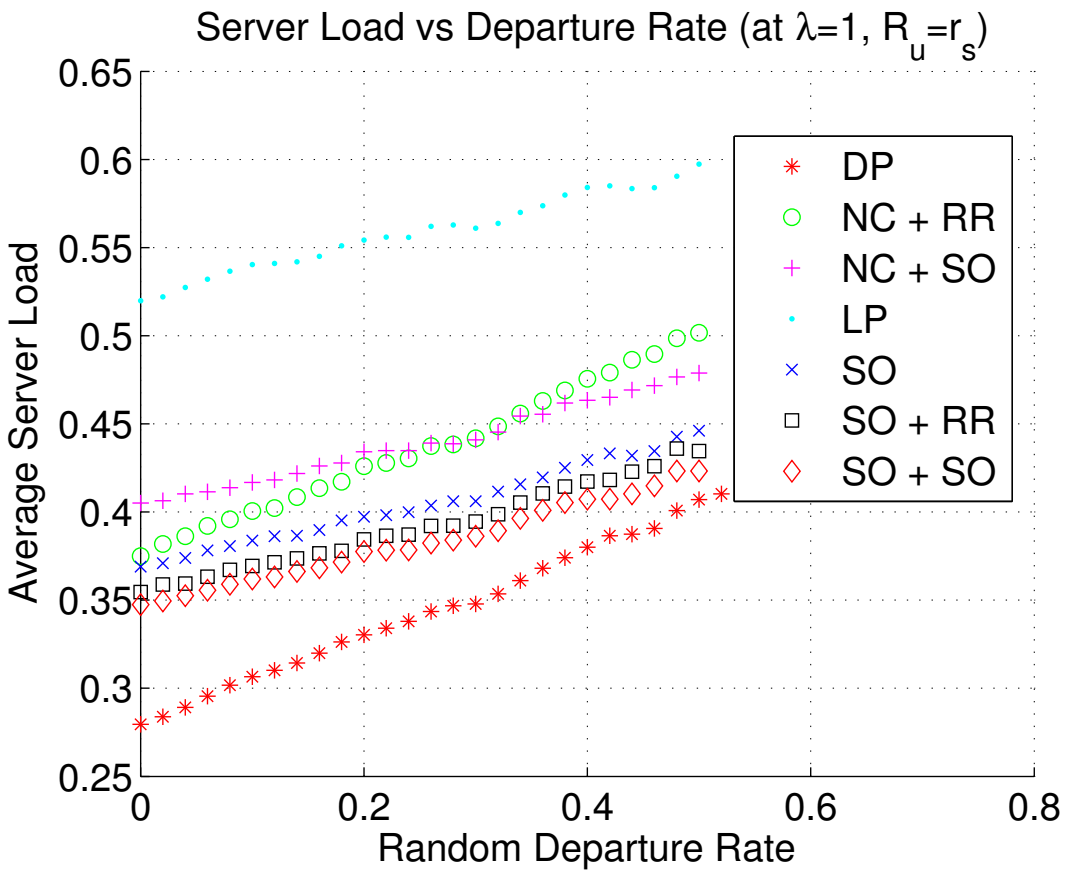

Figure 37: Effect of random early stream termination on normalized server load

compared to the video traffic. Signaling associated with caching decisions occur only once at the beginning of the streaming session for proactive only techniques. The signal exchange periodically occurs for every segment if reactive caching is managed by the control server. However, it is possible to combine caching decision messages with peer queries. Therefore, the extra signaling overhead of reactive methods can be kept similar to proactive methods.

We also overlooked packet losses and transmission delay jitter in our simulations since our focus is on system level performance rather than the end-to-end video quality. Another important factor in peer-to-peer system optimization is the topology of

the created overlay network. System performance can be improved if the users connect to the peers that are geographically close to them. In this thesis study we did not included topology information in the formulation. As a future work, we are planning to improve the efficiency of developed caching methods by taking the network topology into account.

5.5 Conclusions

In this chapter of the thesis we focused on enhancing the scalability of video-on-demand applications to provide low cost streaming service to large number of mobile users. A hybrid architecture where end-systems assist central servers in video distribution is adopted. Users of the service cache segments of the video file and to serve it to their peers who stream the same video. We developed video file segment caching strategies coordinated by a control server. Proposed techniques rely on estimating the future demand and supply of file segments. We defined utility functions to optimize the distribution and schedule of the segments cached by users and targeted to minimize the load of central video servers. Caching methodologies are classified in to two categories. In proactive caching methods, a video segment is pre-fetched by a user when it joins the streaming session. Users only cache and serve file segments that are downloaded in the video streaming process when reactive methods are used. We demonstrated that proposed reactive caching outperforms simple caching techniques that are widely used in the literature. Our reactive caching methods decides the segments to be cached using the dynamic programming solution of the supply-demand utility maximization formulation. Developed methods are tested in scenarios involving flash crowds and premature streaming session termination.

CHAPTER VI

CONCLUSIONS AND FUTURE WORK

6.1 Contributions

In this thesis the challenges of wireless video streaming are addressed in two main categories. *Streaming protocol* level issues constituted the first category.

We developed an application layer streaming protocol improvement that enhances high quality video streaming over wireless home networks. Video rate is dynamically adjusted via transrating to adapt to time-varying wireless bandwidth. A novel technique called *"Delay-constrained and R-D optimized transrating"* is developed. Experimental results are provided to demonstrate the achieved video streaming quality improvement.

We studied the use of scalable coded video for rate-adaptive video streaming over wireless local area networks. New features of the emerging H.264/SVC video coding standard are utilized to develop innovative streaming methods. Results from realistic simulations are presented.

We investigated cross-network layer collaboration techniques for improving real-time multimedia streaming over wireless wide area networks. Forward error correction (FEC) based error protection methods are optimized using video bitstream structure and end-to-end latency constraints. A technique named as *"Finite-Horizon FEC-Rate Adaptation"* is developed.

Finally, we explored streaming service level solutions to reduce the cost of building large scale video-on-demand platforms. Peer-to-peer assisted video streaming technologies are developed to reduce the load of video servers. Novel video file segment caching strategies are proposed for more efficient peer collaboration. Computer

simulation models are constructed to test techniques at diverse set of scenarios.

6.2 Future Research Directions

Our studies on WLAN video streaming focused on tackling the bandwidth inconsistencies of the link via source rate adaptation. The effectiveness of these approaches can be inadequate when the bandwidth degradation last for a long time frame. A solution to this problem is to use multiple transmission links when possible. Fortunately, most of today's WLAN adapters and access points can operate at multiple frequency bands and support different technologies. For instance, dual band interface cards employ both 802.11a (2.4 GHz) and 802.11g (5 GHz) technologies. One of the largest causes of long lasting bandwidth degradations is the signal interference originated from cordless phones or Bluetooth devices. When one of these interfering devices are in use over a frequency band, the quality of the wireless AV stream may be maintained by switching the video sender-client connection to a clearer band or concurrently using multiple bands.

Developed rate-adaptive video streaming methods are optimized for a single-sender single-receiver scenario. Further improvements can be done for scenarios where multiple wireless capable displays are actively streaming video from video gateways.

In the P2P assisted VoD study, the topology of the created overlay network was not optimized. System performance can be improved if the users connect to the peers that are geographically close to them. Proposed caching methods can be modified to take the topology information into account.

REFERENCES

[1] "Bittorrent protocol," *http://www.bittorrent.com/protocol.html*.

[2] "ISO/IEC JTC1/SC 29/WG11 (MPEG), "Joint Scalable Video Model (JSVM) 6", N8015, Montreux, Switzerland, April 2006,"

[3] "Draft amendment to IEEE std. 802.11 (1999 edition), part 11: MAC and PHY specifications: Medium access control (MAC) quality of service (QoS) enhancements," February 2004.

[4] 25.855, G. T. S. G. T., "High speed downlink packet access (HSDPA); overall UTRAN description,"

[5] Androutsellis-Theotokis, S. and Spinellis, D., "A survey of peer-to-peer content distribution technologies," *ACM Comput. Surv.*, vol. 36, no. 4, pp. 335–371, 2004.

[6] Annapureddy, S., Guha, S., Gkantsidis, C., Gunawardena, D., and Rodriguez, P., "Exploring vod in p2p swarming systems," *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pp. 2571–2575, 6-12 May 2007.

[7] Bakre, A. and Badrinath, B. R., "I-TCP: Indirect TCP for mobile hosts," in *15th International Conference on Distributed Computing Systems (ICDCS)*, May 1995.

[8] Begen, A. C. and Altunbasak, Y., "Estimating packet arrival times in bursty video applications," in *IEEE Int. Conf. Multimedia and Expo (ICME)*, 2005.

[9] Cabrera, J., Ortega, A., and Ronda, J., "Stochastic rate-control of video coders for wireless channels," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, pp. 496–510, June 2002.

[10] Chakareski, J. and Chou, P. A., "Application layer error correction coding for rate-distortion optimized streaming to wireless clients," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2002.

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

[12] C.K. Miller, "Multicast Networking and Applications, 1999 Addison Wesley Longman, Reading, MA, ISBN 0-201-30979-3,"

[13] Clark, G., Cain, J., and Geist, J., "Punctured convolutional codes of rate (n-1)/n and simplified maximum-likelihood decoding," *IEEE Transactions on Information Theory*, vol. 25, pp. 97–100, January 1979.

[14] Dan, A., Sitaram, D., and Shahabuddin, P., "Dynamic batching policies for an on-demand video server," *Multimedia Syst.*, vol. 4, no. 3, pp. 112–121, 1996.

[15] Dana, C., Li, D., Harrison, D., and Chuah, C.-N., "Bass: Bittorrent assisted streaming system for video-on-demand," *Multimedia Signal Processing, 2005 IEEE 7th Workshop on*, pp. 1–4, Oct. 2005.

[16] Dempsey, B. J., Liebeherr, J., and Weaver, A. C., "On retransmission-based error control for continuous media traffic in packet-switching networks," *Computer Networks and ISDN Systems*, vol. 28, no. 5, pp. 719–736, 1996.

[17] Deng, R. H. and Lin, M. L., "A type-I hybrid ARQ system with adaptive code rates," *IEEE Transactions on Communications*, vol. 43, no. 2/3/4, pp. 733–737, 1995.

[18] Do, T., Hua, K., and Tantaoui, M., "P2vod: providing fault tolerant video-on-demand streaming in peer-to-peer environment," *Communications, 2004 IEEE International Conference on*, vol. 3, pp. 1467–1472 Vol.3, 20-24 June 2004.

[19] Elaoud, M. and Ramanathan, P., "Adaptive use of error-correcting codes for real-time communication in wireless networks," in *IEEE INFOCOM*, pp. 548–555, 1998.

[20] Farber, N., Girod, B., and Villasenor, J., "Extensions of the ITU-T recommendation H.324 for error resilient video transmission," *IEEE Commun. Mag.*, vol. 36, pp. 120–128, June 1998.

[21] Fritchman, B. D., "A binary channel characterization using partitioned markov chains," *IEEE Transactions on Information Theory*, vol. IT-13 no. 2, pp. 221–227, April 1967.

[22] Fritchman, B. D., "A binary channel characterization using partitioned markov chains," *IEEE Transactions on Information Theory*, vol. IT-13 no. 2, pp. 221–227, 1967.

[23] Girod, B., Kalman, M., Liang, Y. J., and Zhang, R., "Advances in channel-adaptive video streaming," *Wireless Communications and Mobile Computing*, vol. 2, pp. 549–552, September 2002.

[24] Hagenauer, J., "Rate-compatible punctured convolutional codes (RCPC codes) and their applications," *IEEE Transactions on Communications*, vol. 36, pp. 389–400, April 1988.

[25] Hsu, C.-Y., Ortega, A., and Khansari, M., "Rate control for robust video transmission over burst-error wireless channels," *IEEE Journal on Selected Areas in Communications*, vol. 17, May 1999.

[26] Hu, A., "Video-on-demand broadcasting protocols: a comprehensive study," *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 1, pp. 508–517 vol.1, 2001.

[27] Hua, K. A. and Sheu, S., "Skyscraper broadcasting: a new broadcasting scheme for metropolitan video-on-demand systems," *SIGCOMM Comput. Commun. Rev.*, vol. 27, no. 4, pp. 89–100, 1997.

[28] hua Chu, Y., Rao, S. G., and Zhang, H., "A case for end system multicast (keynote address)," in *SIGMETRICS '00: Proceedings of the 2000 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, (New York, NY, USA), pp. 1–12, ACM, 2000.

[29] Hurley, P., Boudec, J., Thiran, P., and Kara, M., "ABE: Providing a low-delay service within best-effort," *IEEE Network*, vol. 15, no. 3, pp. 60–69, 2001.

[30] Ishikawa, E. and de Amorim, C. L., "Cooperative video caching for interactive and scalable vod systems," in *ICN '01: Proceedings of the First International Conference on Networking-Part 2*, (London, UK), pp. 776–785, Springer-Verlag, 2001.

[31] Jain, M. and Dovrolis, C., "End-to-end available bandwidth: Measurement methodology, dynamics, and relation with TCP throughput," *IEEE Trans. on Networking*, vol. 11, pp. 537–549, August 2003.

[32] Johari, R. and Shakkottai, S., "Revenue management for content delivery," *45th Allerton Conference*, September 2007.

[33] Juhn, L.-S. and Tseng, L.-M., "Harmonic broadcasting for video-on-demand service," *Broadcasting, IEEE Transactions on*, vol. 43, no. 3, pp. 268–271, Sep 1997.

[34] Jurca, D., Chakareski, J., Wagner, J.-P., and Frossard, P., "Enabling adaptive video streaming in p2p systems [peer-to-peer multimedia streaming]," *Communications Magazine, IEEE*, vol. 45, no. 6, pp. 108–114, June 2007.

[35] Kang, S. H. and Zakhor, A., "Packet scheduling algorithm for wireless video streaming," in *International Packetvideo Workshop*, 2002.

[36] Kim, T. and Ammar, M., "Optimal quality adaptation for scalable encoded video," *IEEE Journal on Selected Areas in Communications*, vol. 23, pp. 344–356, February 2005.

[37] Kumwilaisak, W., Kim, J., and Kuo, C.-C. J., "Video transmission over wireless fading channels with adaptive FEC," in *Picture Coding Symposium*, pp. 219–222, 2001.

[38] Li, B. and Yin, H., "Peer-to-peer live video streaming on the internet: issues, existing approaches, and challenges [peer-to-peer multimedia streaming]," *Communications Magazine, IEEE*, vol. 45, no. 6, pp. 94–99, June 2007.

[39] Li, Q. and van der Schaar, M., "Providing adaptive QoS to layered video over wireless local area networks through real-time retry limit adaptation," *IEEE Transactions on Multimedia*, vol. 6, pp. 278–290, April 2004.

[40] Li, X., Ammar, M., and Paul, S., "Video multicast over the internet," *Network, IEEE*, vol. 13, no. 2, pp. 46–60, Mar/Apr 1999.

[41] Liu, H. and Zarki, M. E., "Performance of H.263 video transmission over wireless channels using hybrid ARQ," vol. 15, no. 9, pp. 1775–1786, 1997.

[42] MacCarthaigh, C., "Joost network architecture," *7th UK Network Operators Forum*, April 2007.

[43] Miao, Z. and Ortega, A., "Optimal scheduling for streaming of scalable media," in *Asilomar Conf. Signals, Systems, and Computers*, 2000.

[44] Miao, Z. and Ortega, A., "Expected run-time distortion based scheduling for delivery of scalable media," in *Packet Video Workshop*, 2002.

[45] Mukherjee, B. and Brecht, T., "Time-lined TCP for the TCP-friendly delivery of streaming media," 2000.

[46] P. H. Guo, Y. Y. and Li, X. Y., "A p2p streaming service architecture with distributed caching," *Journal of Zhejiang University*, vol. 8, pp. 605–614, April 2007.

[47] Papadopoulos, C. and Parulkar, G. M., "Retransmission-based error control for continuous media applications," *ACM NOSSDAV*, 1996.

[48] Paxson, V., "End-to-end internet packet dynamics," in *Proc. ACM SIGCOMM '97, pp. 139-159, September 1997, Cannes, France.*

[49] Perkins, C., Hodson, O., and Hardman, V., "A survey of packet loss recovery techniques for streaming audio," *IEEE Network*, vol. 12, no. 5, pp. 40–48, 1998.

[50] Prasad, R., Murray, M., Dovrolis, C., and Claffy, K., "Bandwidth estimation: metrics, measurement techniques, and tools," *IEEE Network*, vol. 17, November 2003.

[51] Qiao, D. and Shin, K. G., "A two-step adaptive error recovery scheme for video transmission over wireless networks," in *IEEE INFOCOM*, pp. 1698–1704, 2000.

[52] Quinn, B. and Almeroth, K., "IP multicast applications: challenges and solutions, Internet RFC 3170, Sept. 2001. http://www.ietf.org,"

[53] Reyes, G. D. L., Reibman, A., Chang, S., and Chuang, J., "Error-resilient transcoding for video over wireless channels," *IEEE Transactions on Multimedia*, vol. 18, pp. 1063–1074, June 2000.

[54] S. Annapureddy, C. G. and Rodriguez, P., "Providing video-on-demand using peer-to-peer networks," *Internet Protocol TeleVision (IPTV) Workshop, WWW 06, Edinburgh, Scotland, May 2006*.

[55] Setton, E. and Apostolopoulos, J., "Towards quality of service for peer-to-peer video multicast," *Image Processing, 2007. ICIP 2007. IEEE International Conference on*, vol. 5, pp. V –81–V –84, Sept. 16 2007-Oct. 19 2007.

[56] Stockhammer, T., Hannuksela, M. M., and Wiegand, T., "H.264/AVC in wireless environments," *IEEE Transactions on Circuits and Systems for Video Technology, Special Issue on the H.264/AVC Video Coding Standard*, vol. 13, no. 7, pp. 657–673, 2003.

[57] Stockhammer, T., Jenkac, H., and Kuhn, G., "Streaming video over variable bit-rate wireless channels," *IEEE Transactions on Multimedia*, vol. 6, April 2004.

[58] Tan, W.-T. and Zakhor, A., "Video multicast using layered FEC and scalable compression," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 11, no. 3, pp. 373–386, Mar 2001.

[59] Tang Yun, Y., Luo Jian-Guang, J.-G., Zhang Meng, M., Yang Shi-Qiang, S.-Q., and Zhang Qian, Q., "Deploying p2p networks for large-scale live video-streaming service [peer-to-peer multimedia streaming]," *Communications Magazine, IEEE*, vol. 45, no. 6, pp. 100–106, June 2007.

[60] van Beek, P. and Demircin, M. U., "Delay-constrained rate adaptation for robust video transmission over home networks," in *IEEE Int. Conf. Image Processing (ICIP)*, 2005.

[61] van Beek, P., Deshpande, S., Pan, H., and Sezan, I., "Adaptive streaming of high-quality video over wireless LANs," in *Visual Communications and Image Processing 2004 (VCIP 2004), Proc. SPIE*, vol. 5308, pp. 647–660.

[62] Viswanathan, S. and Imielinski, T., "Metropolitan area video-on-demand service using pyramid broadcasting," *Multimedia Syst.*, vol. 4, no. 4, pp. 197–208, 1996.

[63] Wah, B., Su, X., and Lin, D., "A survey of error-concealment schemes for real-time audio and video transmissions over the internet," in *IEEE Int. Symp. Multimedia Software Engineering*, 2000.

[64] Wang, H. S. and Moayeri, N., "Finite-state markov channel - a useful model for radio communication channels," *IEEE Transactions on Vehicular Technology*, vol. 44, pp. 163–171, February 1995.

[65] Wang, H. S. and Moayeri, N., "Finite-state markov channel - a useful model for radio communication channels," *IEEE Transactions on Vehicular Technology*, vol. 44, pp. 163–171, Feb. 1995.

[66] Wang, Y., Wenger, S., Wen, J., and Katsaggelos, A. K., "Review of error resilient coding techniques for real-time video communications," *IEEE Signal Processing Mag.*, vol. 17, pp. 61–82, July 2000.

[67] Yavatkar, R. and Bhagwat, N., "Improving end-to-end performance of TCP over mobile internetworks," in *Workshop on Mobile Computing Systems and Applications, Santa Cruz, CA*, pp. 146–152, December 1994.

[68] Zhang, M., Luo, J.-G., Zhao, L., and Yang, S.-Q., "A peer-to-peer network for live media streaming using a push-pull approach," in *MULTIMEDIA '05: Proceedings of the 13th annual ACM international conference on Multimedia*, (New York, NY, USA), pp. 287–290, ACM, 2005.

[69] Zhang, Q. and Kassam, S. A., "Finite state markov model for rayleigh fading channels," *IEEE Transactions on Communications*, vol. 47, no. 11, pp. 1688–1692, 1999.

[70] Zhang, Q. and Kassam, S. A., "Finite state markov model for rayleigh fading channels," *IEEE Transactions on Communications*, vol. 47, no. 11, pp. 1688–1692, 1999.

[71] Zhang, X., Liu, J., Li, B., and Yum, Y.-S., "Coolstreaming/donet: a data-driven overlay network for peer-to-peer live media streaming," *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 3, pp. 2102–2111 vol. 3, 13-17 March 2005.

VITA

Mehmet Umut Demircin received his B.S. degree in electrical and electronics engineering in 2001 from Bilkent University, Ankara, Turkey, and the M.S. degree in electrical and computer engineering in 2004 from the Georgia Institute of Technology, Atlanta. He has done his Ph.D. research under the supervision of Prof. Yucel Altunbasak at Georgia Tech. His research areas include robust video coding and streaming over wireless and peer-to-peer networks. He was summer intern in Sharp Laboratories of America, WA in 2004, 2005 and in DoCoMo USA Labs, CA in 2007. He is currently employed as a member of technical staff in the DSPS R&D department of Texas Instruments, Dallas.