ADAPTIVE TRANSMISSION OF VARIABLE-BIT-RATE-VIDEO STREAMS TO MOBILE DEVICES

by

Farid Molazem Tabrizi
B.Sc., Amirkabir University of Technology, Iran, 2005

A Thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science
in the School
of
Computing Science

SIMON FRASER UNIVERSITY
Spring 2011

APPROVAL

Name: Farid Molazem Tabrizi

Degree: Master of Science

Title of Thesis: Adaptive Transmission of Variable-Bit-Rate- Video Streams to Mobile Devices

Examining Committee:

Dr. Arrvind Shriraman,
Assistant Professor of Computing Science
Chair

Dr. Mohamed Hefeeda
Senior Supervisor
Associate Professor of Computing Science

Dr. Joseph G. Peters
Senior Supervisor
Professor of Computing Science

Dr. Jiangchuan (JC) Liu
Examiner
Associate Professor of Computing Science

Date Approved: 22 March 2011

Declaration of
Partial Copyright Licence

Abstract

In recent years, mobile devices have become more powerful in terms of computing power, memory, and screen size. This has resulted in increasing demand for multimedia services for mobile devices. In this thesis, we focus on two research problems of maximizing energy saving and maximizing ontime transmitted video frames in multicast/broadcast mobile video networks. We propose a novel algorithm to efficiently transmit multiple variable-bit-rate video streams from a base station to mobile receivers in wide-area wireless networks. The proposed algorithm transmits video streams in bursts to save the energy of mobile devices and adaptively controls the buffer level of mobile receivers. We have implemented the proposed algorithm as well as two other recent algorithms in a mobile video streaming testbed. Our extensive analysis and results demonstrate that the proposed algorithm outperforms recent algorithms and it results in higher energy saving for mobile devices and fewer dropped video frames.

Keywords: mobile video streaming; mobile tv; energy optimization; quality optimization

Acknowledgments

I am heartily thankful to my supervisors Dr. Mohamed Hefeeda and Dr. Joseph Peters for their patience, guidance, and continuous support from the initial to the final level of my Master studies. I am indebted to them and without their constant guidance and encouragement, I could not have finished this thesis.

I would like to thank all my colleagues at Network Systems Lab. Working with them was a wonderful experience for me and they made the lab environment friendly and convivial.

I was blessed with having some great friends with me during my Master studies. I would like to thank Maryam, Vahid, Pooya, Azadeh, Neda, Sahar, and Behnam. They made my life much easier at the times of difficulties.

My deepest gratitude goes to my family for their unconditional love and support throughout my life. None of my achievements would have been possible without them.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In this chapter, we first overview the challenges related to video streaming applications in mobile networks. Based on this, we define a research problem to improve quality of service in mobile video networks and solve the discussed challenges. Then we summarize our contributions in solving the stated problem and describe the organization of the rest of the thesis.

1.1 Introduction

In recent years, mobile devices have become more powerful in terms of computing power, memory, screen size and screen quality. This has resulted in increasing demand for multimedia services for mobile devices [19]. Also rapid growth of wireless communication and networking protocols has provided necessary infrastructure for growth of video streaming applications for mobile devices. Video streaming provides the opportunity for simultaneous transmission and playback of the video. Compared to file downloading, the users do not have to wait for the entire video to be downloaded. Also since only a small portion of the video is saved in the buffer before being played out, less memory is required for the mobile devices to access and watch multimedia content.

Although video streaming services could be provided through unicast and over cellular networks, these services are very demanding in terms of network resources. Unicast streaming creates a one-to-one connection between the server and the client and each client receives a distinct video stream from the server. Establishing a separate connection for each client will quickly overwhelm bandwidth resources as the number of unicast clients

increases. In broadcast/multicast streaming, the same video data is transmitted to multiple clients at the same time. This makes broadcast and multicast streams much less demanding on network resources. Based on this, the need for multicast/broadcast services to support multimedia transmission is clear and in our thesis, we focus on video streaming over wireless broadcast networks. Broadcast services have been addressed in wireless networks including WiMax [45], DVB-H (Digital Video Broadcast-Handheld) [28], MediFLO [18], and Multimedia Broadcast Multicast Service (MBMS) [35].

Video streaming to mobile devices still has many challenges that need to be addressed. For example, mobile devices are small and can only be equipped with small batteries that have limited lifetimes. Thus, conserving the energy of mobile devices during streaming sessions is needed to prolong the battery lifetime and enable users to watch videos for longer periods. Therefore, energy saving is an important quality of service metric for mobile video services. Some broadcast networks like DVB-H and MediaFLO transmit video data in bursts to let the receivers save energy. This technique is called time slicing. Based on this, the server does not transmit video data continuously, but in compact bursts with higher bit rate than the actual bit rate of the video streams. This lets the receivers of each video stream to be active only during short time periods to receive video bursts and go into energy conservation mode during the time interval between receiving consecutive bursts.

Another challenge for mobile video is the limited wireless bandwidth in wide-area wireless networks. The wireless bandwidth is not only limited, but it is also quite expensive. For instance, Craig Wireless System Ltd. agreed to sell one quarter of its wireless spectrum to a joint venture of Rogers Communication and Bell Canada for $80 million [7], and AT&T sold a 2.5 GHz spectrum to Clearwire Corporation in a $300 million transaction [5]. Thus, for commercially viable mobile video services, network operators should maximize the utilization of their license-based wireless spectrum bands. Maximizing bandwidth utilization means maximizing the number of video streams that can be concurrently transmitted while having the rate of frame loss within an acceptable range. We consider a frame as a lost frame or a dropped frame if it is not transmitted to the receiver before its decoding deadline. Otherwise, we call it an ontime frame. In this regard, minimizing the number of lost frames is equivalent to maximizing the number of ontime transmitted frames. Frame loss rate is an important quality of service metric in video streaming systems. A high rate of frame loss will result in screen freezes and playout glitches which cause user dissatisfaction and poor watching experience for viewers.

Some video encoders generate constant bit rate (CBR) streams to simplify the allocation of network resources. Since different frames are different in the level of complexity, enforcing constant bit rate to the video stream will result in uneven quality of the video which is not desirable for viewers. On the other hand, variable bit rate (VBR) encoding of video streams produces video with constant and higher quality, but produces challenging resource allocation problems in video transmission applications. Based on the advantages of VBR encoding, we consider VBR video streams in this thesis.

In this thesis, we consider the problem of multicasting multiple variable bit rate (VBR) video streams from a wireless base station to many mobile receivers over a common wireless channel. This problem arises in wide-area wireless networks that offer multimedia content using multicast and broadcast services, such as DVB-H [28], ATSC M/H (Advanced Television Systems Committee-Mobile/Handheld) [4], WiMAX [45], and 3G/4G cellular networks that enable the Multimedia Broadcast Multicast Services [35].

We propose a new algorithm to efficiently transmit the video streams from the base station to mobile receivers. The goal of our algorithm is to minimize the number of lost frames and maximize the energy saving of mobile devices.

1.2 Problem Statement and Thesis Contributions

In this, thesis we study optimization of two important quality metrics for mobile video streaming services: energy saving and rate of frame loss. Our first objective is to minimize the number of lost frames when transmitting variable bit rate video streams from a base station to mobile receivers with limited buffer capacity. Our second objective is to maximize energy saving of the mobile receivers by transmitting video data in bursts.

1.2.1 Problem Statement

Frame loss rate is an important metric to evaluate the quality of a video streaming service. In video streaming, a frame is lost if it is not transmitted to the receivers before its decoding deadline. Wireless bandwidth is limited and expensive. Therefore, service providers have to employ mechanisms to optimize the use of their licensed bandwidth. Developing algorithms to minimize frame loss rate will allow the operators to provide high quality service and at lower costs. But achieving this objective when transmitting VBR video streams

is a challenging problem. In VBR streams, the aggregate bit rate of video streams could instantaneously increase which makes a fixed resource allocation policy inefficient.

Mobile devices carry small batteries and thus energy conservation is another important concern for mobile applications. Therefore, video streaming techniques should be designed in a way to account for this limitation in order to be efficiently deployed for mobile networks. A burst scheduling technique is used to let handheld devices save energy during a streaming session. In this technique, video data is not transmitted continuously, but in bursts with higher bit rate than the actual bit rate of the video stream. Therefore, receivers can be active for the period of receiving bursts, save the bursts into their buffers and then go to energy conservation mode when there is no burst to receive. In this thesis, we consider the average energy saving achieved among all video streams as our energy saving performance metric.

We state the problem we need to solve in the following.

Problem. *Create a burst transmission schedule for broadcasting a set of variable bit rate video streams from a base station to mobile devices such that: (i) the number of video frames received before their decoding deadline is maximized, (ii) the average energy saving among all video streams is maximized, and (iii) no buffer overflow and underflow instances occur at the receivers.*

1.2.2 Thesis Contributions

In this thesis we study and solve the problem of transmitting a set of video streams from a base station to mobile devices while maximizing ontime transmitted frames and energy saving. Our contributions can be summarized as follows [33, 42].

- We propose a polynomial, dynamic burst scheduling algorithm for multiplexing variable bit rate video streams to maximize ontime transmitted frames while achieving high energy saving for mobile devices. Unlike previous algorithms, the new algorithm adaptively controls the buffer level of mobile devices receiving different video streams. This is done through dynamic control of the wireless bandwidth. Our proposed algorithm uses variable bit rate video streams, which are statistically multiplexed to increase the utilization of the expensive wireless bandwidth. We analytically prove that our algorithm can always find a feasible schedule if there exists one.

- We use burst scheduling technique in our algorithm to achieve energy saving for mobile receivers. It has been shown that creating an optimal burst schedule is an NP-Complete problem [23]. We show that our algorithm is polynomial and analytically prove that the gap between energy saving resulted from our algorithm and optimal energy saving is bounded by a small value.
- We implement our new algorithm in a mobile video streaming testbed to practically demonstrate its performance. The results from the testbed confirm that our algorithm runs in real-time and achieves high performance in terms of energy saving and transmitting video frames ontime.

1.3 Thesis Organization

The rest of this thesis is organized as follows. We provide an overview of multicast/broadcast networks for multimedia streaming and discuss the related research to address the challenges in this domain in Chapter 2. In Chapter 3 we describe and analyze our proposed solution for the optimization problems of maximizing ontime transmitted frames and maximizing energy saving for mobile receivers. We extensively evaluate our algorithm and provide the results in Chapter 4. We conclude the thesis in Chapter 5 and describe the directions in which this work could be extended in the future.

Chapter 2

Background

In this chapter, we overview video broadcast/multicast networks and describe their characteristics. We also discuss current algorithms proposed in the literature and used in practice to solve challenges related to transmitting variable bit rate video streams in mobile networks.

2.1 Video Broadcast Standards

Broadcast/Multicast networks, compared to unicast services, use network infrastructure more efficiently as the source is required to send data only once even if it needs to be received by a large number of receivers. Broadcasting and multicasting multimedia may be performed through cellular networks, WiMAX (Worldwide Interoperability for Microwave) networks, or dedicated broadcast networks such as DVB-H [13,15,28] and MediaFLO [8,18]. Traditional cellular networks only support unicast which puts a large overhead on bandwidth resources and is not efficient for multimedia applications. Therefore, 3rd Generation Partnership Project (3GPP) [3] has proposed Multimedia Broadcast and Multicast Services (MBMS) for efficient utilization of cellular network resources in multimedia applications. MBMS is a broadcasting service using existing GSM (Global System for Mobile Communications) [21] and UMTS (Universal Mobile Telecommunications System) [2] technologies. This service provides an uplink channel that facilitates interaction between user and service provider. The main use of MBMS service is Mobile TV. It supports two modes, broadcast, which provides unidirectional point-to-multipoint transmission of multimedia, and multicast, which generally needs subscription to the multicast group. The WiMAX standard is specified by the IEEE 802.16 standard [26] and is a wireless technology that provides

high throughput broadband connection over long distance. WiMAX base stations can cover up to 45 kilometers and provide wireless metropolitan area network (MAN) connectivity. WiMAX can support high peak data rates of up to 40Mbps bandwidth. The physical layer in WiMAX is based on Orthogonal Frequency Division Multiplexing (OFDM) technology which is resistant to multipath interference. WiMAX supports different modulation and Forward Error Correction (FEC) coding schemes that can be changed based on channel conditions. This adaptive modulation and coding (AMC) is very useful for enhancing throughput in a time-varying channel. WiMAX is a bidirectional technology and this makes it suitable for interactive multimedia applications.

There are several dedicated mulimedia broadcast network standards including MediaFlo [8, 18], and DVB-H [13, 15, 28]. The MediaFLO standard has been developed by Qualcomm technology for mobile multimedia multicast applications. A new air interface has been developed for the MediaFLO standard based on FLO (Forward Link Only) technology. This technology is built from scratch and hence, it is not facing any backward compatibility constraints. MediaFLO systems consist of two parts, the FLO network which includes transmitters and backhaul network, and the FLO device which includes any mobile or handheld FLO-enabled device. FLO physical layer targets transmission in the VHF/UHF/L-band frequency bands and over channel bandwidths of 5, 6, 7, and 8 MHz. DVB-H [13, 15, 28] is another standard for transmission of braodcast content to handheld devices. It was developed by international DVB (Digital Video Broadcasting Project) and published by ETSI (European Telecommunication Standards Institute). DVB-H was developed to support the special requirements of handheld devices with limited battery capacity, and is built based on the DVB-T [12] standard for digital terrestrial television which is already deployed in many countries around the world. DVB-H employs a time slicing scheme in order to achieve energy saving for mobile receivers. In the DVB-H standard, a Forward Error Correction (FEC) stage is added to enhance the reception of mobile devices in different areas.

An important technique to support energy saving in mobile multimedia networks is time slicing. Time slicing is supported by some network standards such as MediaFLO and DVB-H. In this scheme, service multiplexing is performed through time division multiplexing. This enables receivers to process only that part of the stream that is related to the current service selected by the receiver. Based on this, data of a service is not transmitted continuously, but in compact bursts with gaps in between. In other words, data is transmitted in bursts

with significantly higher bit rates than the actual bit rate of video streams. The received bursts will be stored in the buffers of mobile devices. Therefore, services are processed by the receivers in a time-selective manner. Devices are synchronized to receive the bursts of the service they need and go into energy conservation mode in the time intervals between receiving bursts. For each service, the position of each burst is computed based on the relative time difference of two consecutive bursts. This time difference is included in the header of the proceeding burst so that the receiver could be synchronized with the burst transmission schedule. Considering the fact that handheld devices might be moved from one place to another place on a regular basis, time slicing can also result in soft handover if the receivers move from one network cell to another network cell.

2.2 Related Work

Energy saving at mobile receivers using burst transmission (aka time slicing) has been studied in [14, 46]. Simulations are used in these studies to show that time slicing can improve energy saving for mobile receivers. However, no burst transmission algorithms are presented. Some video encoders adjust the quality of video streams to produce constant bit rate video streams which makes them less complex when being transmitted. But this results in noticeable quality degradation for the video [16, 29]. A burst transmission algorithm for constant-bit-rate (CBR) video streams is proposed in [25]. This algorithm can not handle VBR video streams with fluctuating bit rates. In this thesis, we consider the more general VBR video streams which can achieve better visual quality and bandwidth utilization [31].

Variable-bit-rate encoding of video streams results in higher quality video by assigning more bits to complex frames and fewer bits to less complex frames. This results in more complicated requirements for applications of VBR streams [20]. Transmitting VBR video streams over a wireless channel while avoiding buffer overflow and underflow at mobile devices is a difficult problem [37]. Rate smoothing is one approach to reducing the complexity of this problem. The main idea in this approach is to transmit video streams in segments of constant bit rate and let the receiver buffer store some data ahead of time to avoid large changes in the transmission bit rate of video streams. This reduces the complexities resulting from variability of video stream bit rates. The performance of smoothing algorithms could be measured via different metrics like the peak bit rate, the number of changes in the bit rate, traffic variability and the receiver buffer requirement. The minimum requirements

of rate smoothing algorithms in terms of playback delay, lookahead time, and buffer size are discussed in [44]. The smoothing algorithms proposed in [9, 17, 47] minimize the number of rate changes.

In [36, 40] a smoothing algorithm is proposed to minimize traffic variability subject to a given receiver buffer and startup delay. The algorithms in [32] reduce the complexity by controlling the transmission rate of a single video stream to produce a constant bitrate stream. The authors of [39, 41] suggest smoothing algorithms to reduce the peak bandwidth requirements for video streams. In [30], a monotonically decreasing rate (MDR) algorithm for smoothing the bit rates of video streams is discussed. This algorithm produces segments with monotonically decreasing bit rate for video streams to remove the resource allocation complexities resulting from upward adjustment. The worst case buffer requirement in this algorithm is unbounded which makes it unsuitable for mobile receivers with limited buffer capacity. None of the above smoothing algorithms considers energy saving as a performance metric as these algorithms are not designed for mobile multicast/broadcast networks with limited-energy receivers. In [6], an online smoothing algorithm for bursts (SAB) is introduced which tries to minimize the percentage of lost frames due to bandwidth and buffer limitations. The algorithm calculates the minimum and maximum possible bit rates for video stream without experiencing buffer underflow and overflow instances. This algorithm transmits video data in bursts to achieve energy saving for mobile receivers. SAB, however, considers the transmission of only one video stream.

A different approach to handling VBR video streams is to employ joint rate control algorithms. In this approach, controlling the bit rates of video streams is achieved by using joint video coders [22, 27, 38, 43], which consist of joint rate allocators, VBR coders and decoders to jointly encode video streams and dynamically allocate bandwidth to them. For example, Rezaei et al. [38] propose joint video coding and statistical multiplexing for video transmission. In their work, statistical multiplexing is implemented inside the encoders so the bandwidth is distributed among the bitstreams according to their coding complexities. They use time slicing to achieve energy saving. But this mechanism cannot be used if we do not have access to expensive joint video encoders.

A recent algorithm for transmitting VBR video streams to mobile devices without requiring joint video encoders is presented in [24]. This algorithm, called SMS, performs statistical multiplexing of video streams. Unlike the proposed algorithm in this thesis, the SMS algorithm does not dynamically control the buffers of the receivers. We show that

this dynamic control of buffers improves the performance in terms of dropped video frames and energy consumption of mobile devices. The VBR transmission algorithms deployed in practice are simple heuristics. For example, in the Nokia Mobile Broadcast Solution (MBS) [1, 34], the operator determines a bitrate value for each video stream and a time interval based on which bursts are transmitted. The time interval is calculated on the basis of the size of the receiver buffers and the largest bitrate among all video streams and is used for all video streams to avoid buffer overflow instances at the receivers. In each time interval, a burst is scheduled for each video stream based on its bitrate. It is difficult to assign bitrate values to VBR video streams to achieve good performance while avoiding buffer underflow and overflow instances at the receivers. In this thesis, we compare our proposed algorithm to the SMS algorithm [24] (which represents the state-of-the-art in the literature) as well as to the algorithm used in the Nokia Mobile Broadcast Solution (which represents one of the state-of-the-art algorithms in practice).

Chapter 3

Optimized Mobile Video Transmission

In this chapter, we describe our proposed algorithm for transmission of video streams from a base station to mobile receivers while maximizing energy saving and maximizing ontime transmitted frames. First, we define the system model we use in this thesis based on which we design our solution. Then, we describe our algorithm, Adaptive Data Transmission (ADT) algorithm, to address the problem of lost frames and energy saving in mobile video networks.

3.1 System Model

We study the problem of transmitting several video streams from a wireless base station to a large number of mobile receivers. We focus on multicast and broadcast services enabled in many recent wide-area wireless networks such as DVB-H [28], MediaFLO [18], WiMAX [45], and 3G/4G cellular networks that offer Multimedia Broadcast Multicast Services (MBMS) [35]. In such networks, a portion of the wireless spectrum can be set aside to concurrently broadcast multiple video streams to many mobile receivers. Since the wireless spectrum in wide-area wireless networks is license based and expensive, maximizing the utilization of this spectrum is important. To achieve high bandwidth utilization, we employ the variable-bit-rate (VBR) model for encoding video streams. Unlike the constant-bit-rate (CBR) model, the VBR model allows statistical multiplexing of video streams [24], and yields

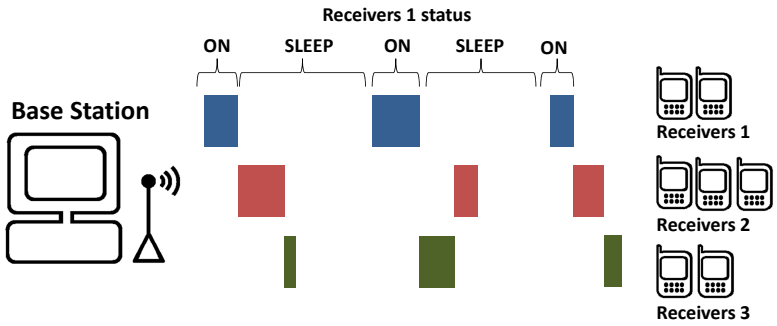


Figure 3.1: The wireless network model considered in this thesis.

better perceived video quality [31]. However, the VBR model makes video transmission much more challenging than the CBR model in mobile video streaming networks [37].

Mobile receivers are typically battery powered. Thus, reducing the energy consumption of mobile receivers is essential. To save energy, we employ the burst transmission model for transmitting video streams, in which the base station transmits the data of each video stream in bursts with a bit rate higher than the encoding bit rate of the video. The burst transmission model allows a mobile device to save energy by turning off its wireless interface between the reception of two bursts [23, 37]. The arrival time of a burst is included in the header of its preceding burst. Thus, the clocks at mobile receivers do not need to be synchronized. In addition, each receiver is assumed to have a buffer to store the received data. Figure 3.1 shows a high-level depiction of the system model that we consider.

We note that in wireless broadcast/multicast networks, the parameters of the physical layer, e.g., signal modulation and transmission power, are fixed for all receivers. These parameters are chosen to ensure an average level of bit error rate for all receivers in the coverage area of the base station. In addition, in such networks, the transmission is one-way from the base station to the receivers, with no feedback channel from the receivers since there could be many receivers. However, an external channel can exist to allow user interactivity with the transmitted video streams, for example voting for best player during a soccer game. The external channel is a unicast channel and our work does not optimize it. We optimize the downward broadcast/multicast channel.

We list all symbols used in our formulation in Table 3.1.

Symbol	Description
α	Control parameter indicating the fraction of buffer that should be played out before reaching the next control point
B	receiver buffer size (Kb)
b_k^s	Size of burst k of stream s (Kb)
d_s	Playout deadline for receivers of stream s
F	Frame rate of video streams (frame-per-sec)
f_k^s	Start time of burst k of stream s (sec)
g_s	Number of frames of stream s that are be played out between current control point and next one
Γ	Size of scheduling window
γ	Average energy saving among all video streams
γ_s	Energy saving for receivers of stream s
h_s	Time of the next control point for video stream s (sec)
I	Total number of frames in a video stream
l_i^s	Size of frame i of stream s (Kb)
M	Maximum burst size (Kb)
m_s	Number of frames sent to receivers of stream s
N	Total number of control points among all video streams
p_s	Number of frames played out at receivers of stream s
R	Available bandwidth (Kbps)
r_s	Average bitrate of video stream s (Kbps)
S	Number of video streams
T	Length of each video stream (sec)
T_o	Wake-up time for the receiver circuit (sec)

Table 3.1: Symbols used in the thesis

3.2 Problem Statement

To achieve the burst transmission of video streams described in Figure 3.1, we need to create a transmission schedule which specifies for each stream the number of bursts, the size of each burst, and the start time of each burst. Note that only one burst can be transmitted on the broadcast channel at any time. The problem we address in this section is to design an algorithm to create a transmission schedule for bursts that yields better performance than current algorithms in the literature.

In particular, we study the problem of broadcasting S VBR video streams from a base station to mobile receivers in bursts over a wireless channel of bandwidth R Kbps. The base station runs the transmission scheduling algorithm every Γ sec; we call Γ the scheduling

window. The base station receives the video data belonging to video streams from streaming servers and/or reads it from local video databases. For example, the base station can be broadcasting a live hockey game, which it receives from the streaming server of a sports media network, as well as several pre-recorded TV episodes and movies from a video database. The base station aggregates video data for Γ sec. Then, it computes for each stream s the required number of bursts. We denote the size of burst k of video stream s by b_k^s (Kb), and the transmission start time for it by f_k^s sec. The end time of the transmission for burst k of stream s is $f_k^s + b_k^s/R$ sec.

After computing the schedule, the base station will start transmitting bursts in the next scheduling window. Each burst may contain multiple video frames. We denote the size of frame i of video stream s by l_i^s (Kb). Each video frame i has a decoding deadline, which is i/F, where F is the frame rate (fps). The goals of our scheduling algorithm are: (i) maximize the number of frames delivered on time (before their decoding deadlines) for all video streams, and (ii) maximize the average energy saving for all mobile receivers. We define the average energy saving as $\gamma = \sum_{s=1}^{S} \gamma_s/S$, where γ_s is the fraction of time the wireless interfaces of the receivers of stream s are turned off.

3.3 Proposed Algorithm

3.3.1 Overview

We propose a novel algorithm, which we call the Adaptive Data Transmission (ADT) algorithm, to solve the burst transmission problem for the VBR video streams described in Section 3.1. The key idea of the algorithm is to *adaptively* control the buffer levels of mobile devices receiving different video streams. The buffer level at each mobile device is adjusted as a function of the bit rate of the video stream being received by that device. Since we consider VBR video streams, the bit rate of each video is changing with time according to the visual characteristics of the video. This means that the buffer level at each mobile device is also changing with time. The receiver buffer level is controlled through the size and timing of the bursts transmitted by the base station in each scheduling window. The size and timing of the transmitted bursts are computed by the proposed ADT algorithm. The adaptive control of the receiver buffers provides flexibility to the base station, which can be used to transmit more video data on time to mobile receivers.

The ADT algorithm defines control points at which it makes decisions about which

stream should have access to the wireless medium and for how long is should have access. Control points for each video stream are determined separately, and based on a parameter denoted by α, where $0 < \alpha \leq 1$. This parameter is the fraction of a receiver's buffer B that is played out between two control points. The parameter α can change dynamically between scheduling windows but is the same for all video streams. At a given control point, the base station selects a stream and computes the buffer level of the receivers for the selected stream and can transmit data as long as there is no buffer overflow at the receivers.

For small values of α, control points are closer to each other (in time) which results in smaller bursts. This gives the base station more flexibility on deciding which video stream should be transmitted to meet its deadline. That is, the base station has more opportunities to adapt to the changing bit rates of the different VBR video streams being transmitted. For example, the base station can quickly transmit more bursts for a video stream experiencing high bit rate in the current scheduling window and fewer bursts from another stream with low bit rate in the current scheduling window. This dynamic adaptation increases the number of video frames that meet their deadlines from the high-bit rate stream while not harming the low-bit rate stream. However, smaller bursts may result in less energy saving for the mobile receivers because they may turn their wireless interfaces on and off more often. In each transition from off to on, the wireless interface incurs an overhead because it has to wake up shortly before the arrival of the burst to initialize its circuits and lock onto the radio frequency of the wireless channel. We denote this overhead by T_o, which is on the order of msec depending on the wireless technology. We analyze the impact of α on the energy saving and number of frames that arrive on time, theoretically in Section 3.4, and empirically using a mobile video streaming testbed in Chapter 4.

3.3.2 Details

The proposed ADT algorithm is to be run by the wireless base station to schedule the transmission of S video streams to mobile receivers. The algorithm can be called periodically every scheduling window of length Γ sec, and whenever a change in the number of video streams occurs. As will be shown in the next section, the algorithm is computationally efficient and can easily run in real time.

We define several variables that are used in the algorithm. Each video stream s is coded at F fps. We assume that mobile receivers of video streams have a buffer capacity of B Kb. We denote the size of frame i of video stream s by l_i^s (Kb). We denote the time by

which the data in the buffer of a receiver of stream s is completely played out by d_s sec. This means that by the time d_s sec, the buffers of receivers of video stream s are completely drained. Thus, d_s is the deadline for stream s for receiving another burst. We use the parameter M (Kb) to indicate the maximum size of a burst that could be scheduled for a video stream in our algorithm when there are no other limitations like buffer size. In some wireless network standards, there might be limitations on the value of M. In this thesis, we assume that the buffer capacity is small, so the upper bound M on the buffer size does not affect our algorithm in practice. A control point is a time when the scheduling algorithm decides which stream should be assigned a burst. A control point for video stream s is set every time the receiver of stream s has played out αB Kb of video data.

The algorithm schedules bursts in scheduling windows of Γ sec. Based on this, in each scheduling window, the scheduler schedules Γ sec. of data that will be played within that scheduling window. We assume that at each scheduling window, the algorithm has access to a small amount of data (as large as the size of the buffer) from the next scheduling window too. Therefore, if all Γ sec. of video data for the video streams were scheduled within the current scheduling window and there is still some extra time remaining, the scheduler can go ahead and continue scheduling some bursts from the next scheduling window within the current scheduling window. Let us assume that the algorithm is currently computing the schedule for the time window t_{start} to $t_{start} + \Gamma$ sec. The algorithm defines the variable $t_{schedule}$ and sets it equal to t_{start}. Then, the algorithm computes bursts one by one and keeps incrementing $t_{schedule}$ until it reaches the end of the current scheduling window, i.e., $t_{start} \leq t_{schedule} \leq t_{start} + \Gamma$. For instance, if the algorithm schedules a burst of size 125 Kb on a 1 Mbps bandwidth, then the length of this burst will be 0.125 sec and the algorithm increments $t_{schedule}$ by 0.125 sec. The number of video frames for video stream s which belong to the current scheduling window and are scheduled until $t_{schedule}$ is denoted m_s. This gives the receiver of stream s, the playout time of m_s/F sec. Based on this, we can define the playout deadline for stream s at time $t_{schedule}$ as:

$$d_s = t_{start} + m_s/F. \tag{3.1}$$

We let p_s denote the number of frames played out at the receivers of stream s by the time $t_{schedule}$. Therefore, we can define p_s as $p_s = t_{schedule} \times F$. The next control point h_s of stream s after time $t_{schedule}$ will be when the receivers of stream s have played out αB Kb of video data. We compute the number of frames g_s corresponding to this amount as

follows:

$$\sum_{i=p_s+1}^{p_s+g_s} l_i^s \leq \alpha B < \sum_{i=p_s+1}^{p_s+g_s+1} l_i^s. \tag{3.2}$$

The control point h_s is then given by:

$$h_s = t_{schedule} + g_s/F. \tag{3.3}$$

Adaptive Data Transmission (ADT) Algorithm

```
1.  // Input: S VBR video streams
1.  // Output: Burst schedule to transmit S video streams
1.  compute control points and deadlines for each stream s
2.  while there is a video stream to transmit {
3.    create a new scheduling window from t_start to t_start + Γ
4.    while the current scheduling window is not complete and α
4.    could be updated through binary search
5.      pick video stream s having the earliest deadline
6.      schedule a burst until the next control point or until the buffer is full
7.      update t_schedule based on the length of scheduled burst
8.      //gradually increase α if it was reduced in the previous scheduling window
9.      if α < α_max and α was not reduced during the current scheduling window
10.       update α (increase linearly)
11.     update d_s and h_s (control point) for video stream s
12.     if there is a stream s' which is late and α > α_min or if α
12.     could still be updated through binary search
13.       //go back within the scheduling window and reschedule bursts
14.       reset t_schedule to t_start
15.       update α through binary search
18.   }
19.  }
```

Figure 3.2: The proposed transmission scheduling algorithm.

In our algorithm, we define α_{min} and α_{max} as the range based on which the value of α can change. The operator presets these values based on the desired control over bandwidth and flexibility in energy saving. The high-level pseudo-code of the ADT algorithm is given in Figure 3.2. The algorithm works as follows. The scheduler picks a new video stream to assign a burst to at two points. First, if there is a control point and second, when the buffer of the receiver of the currently scheduled burst is full (after scheduling the current

burst). When deciding to select a stream, the algorithm finds the stream s' which has the closest deadline $d_{s'}$. Then it finds the stream s'' with the closest control point $h_{s''}$. Then the algorithm schedules a burst for stream s' until the control point $h_{s''}$ if this does not exceed the available buffer space at the receivers of s'. Otherwise, the size of the new burst is set to the available buffer space. If the scheduled burst is as large as the available buffer space, which makes the buffer full, then we do not schedule any burst for stream s' before it's next control point. This lets the algorithm not schedule small bursts for video streams when the buffers are nearly full and results in achieving more energy saving for the receivers. The algorithm repeats the above steps until there are no more bursts to be transmitted from the video streams in the current scheduling window, or until $t_{schedule}$ exceeds $t_{start} + \Gamma$ and there remains data to be transmitted. The latter case means that some frames will not be transmitted. In this case, the algorithm tries to find a better schedule by increasing the number of possible control points to introduce more flexibility. This is done by decreasing value of α. In order to find the largest value of α that gives us the possibility to transmit all video streams, we do binary search between α_{min} and α_{max}. In order to do this, we consider all the values between α_{min} and α_{max} with a constant step (in our case 0.05). This gives us a constant number of choices for values of α. At each step of the binary search, we select a value for α, and run the algorithm. If it was successful, we continue binary search to choose a larger α. This gives us the chance to see if we can succeed with a larger α or not. Otherwise, we continue binary search to choose a smaller α. This process helps us find the largest value of α that lets us successfully schedule all video data within the current scheduling window. If no such α is found, then the binary search will result in selecting α_{min}. It is important to note that since the search space is of the constant size, the number of steps in the binary search is constant too. Hence, the scheduling algorithm will be run a constant number of times to find the best α. If α is reduced in a scheduling window, then it will be gradually increased in the following scheduling windows based on a linear function. This means that after scheduling every burst, the value of α will be increased by a constant value. In our work, we have used 0.01 as this constant value.

3.4 Analysis and Complexity of the ADT algorithm

In this section, we show that the proposed algorithm produces near optimal schedules in terms of energy saving for mobile receivers. We know that the burst scheduling problem

has been shown to be NP-Complete [23]. Thus, solving it optimally and in real time is not feasible. In addition, we show that the proposed algorithm is efficient and it can run in real time. Finally, we prove that the algorithm results in correct burst transmission schedules without making mobile receivers suffer from buffer overflow and underflow instances as well as no two bursts overlap in time.

The following theorem shows that our algorithm produces near optimal energy saving for mobile devices.

Theorem 1. *The difference between the optimal energy saving and the energy saving of the ADT algorithm is less than or equal to* $\frac{\bar{r}T_o}{B}(2/\alpha_{min} - 1)$*, where* $\bar{r}$ *is the average bitrate of all video streams,* T_o *is the wake-up overhead for the mobile receiver circuits,* B *is the buffer size, and* α_{min} *is the minimum value of the control parameter used in the ADT algorithm.*

Proof. The amount of energy saving is related to the number of the bursts scheduled for the video streams. In our algorithm, we schedule a new burst for a video stream whenever we reach a control point or when we schedule a burst for a video stream that completely fills its receiver buffer and we select a new stream to schedule a burst for. So, the number of bursts will be bounded by the number of control points and the number of times a receiver buffer gets full. We can see from the algorithm that for a video stream s, we will not have a new control point unless we have transmitted at least αB bytes of data since the previous control point for s. Since the value of α can vary, we use $\alpha = \alpha_{min}$ in the remainder of this proof to get bounds. This means that the number of control points is at most $r_sT/(\alpha B)$ where r_s is the average bit rate of video stream s and T (sec) is the total length of the video stream. Note that r_sT is the total number of bits in video stream s. When we schedule a burst that fills the buffer of video stream s, we do not schedule a burst for that stream again before its next control point. Based on this, we can see that between two consecutive control points for stream s, we can only have one instance that its receiver buffer gets full and consequently, the total number of these instances will be bounded by the total number of control points. Therefore, the total number of bursts for stream s is at most $2r_sT/(\alpha B)$. Hence, the total overhead time that a wireless receiver of stream s is active is at most $2T_or_sT/(\alpha B)$. The wireless receiver of s will also be active during the data transmission time of stream s which is r_sT/R. Consequently, the total time that a wireless receiver of stream s is active is at most $2T_or_sT/(\alpha B) + r_sT/R$. Assuming that all video streams have the same length T, the total time that the wireless receivers of all video streams are active is less than or equal to

$2T_o \sum_{s=1}^{S} r_s T/(\alpha B) + \sum_{s=1}^{S} r_s T/R$. Using $\bar{r} = \sum_{s=1}^{S} r_s/S$ as the average bitrate among all video streams, we get a lower bound on the energy saving, γ, of the ADT algorithm:

$$\gamma \geq \frac{\sum_{s=1}^{S} T - 2T_o \sum_{s=1}^{S} r_s T/(\alpha B) - \sum_{s=1}^{S} r_s T/R}{\sum_{s=1}^{S} T} = 1 - \bar{r}(\frac{2T_o}{\alpha B} + \frac{1}{R}).$$

The size of a burst cannot exceed the size of the buffer, so the minimum number of bursts that could be scheduled for video stream s is $r_s T/B$. Consequently, the overhead time for a wireless receiver of stream s is at least $T_o r_s T/B$ and the total time that a wireless receiver of stream s is active is greater than or equal to $T_o r_s T/B + r_s T/R$. This gives an upper bound on the optimal energy saving, γ_{opt}:

$$\gamma_{opt} \leq \frac{\sum_{s=1}^{S} T - T_o \sum_{s=1}^{S} r_s T/B - \sum_{s=1}^{S} r_s T/R}{\sum_{s=1}^{S} T} = 1 - \bar{r}(\frac{T_o}{B} + \frac{1}{R}).$$

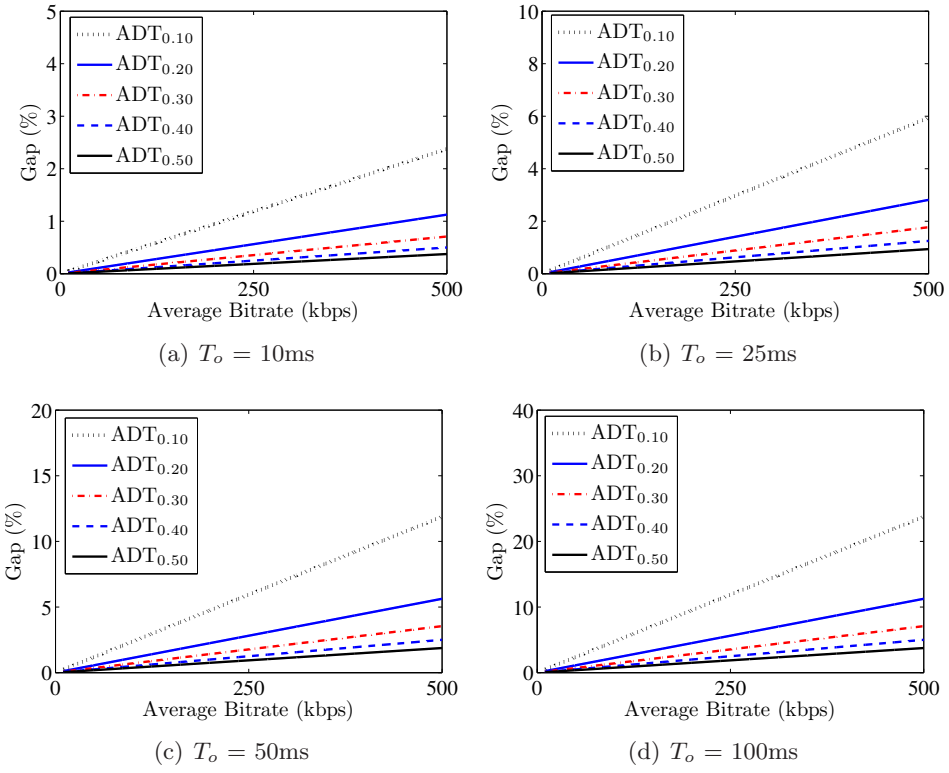
The difference between the optimal energy saving and the energy saving of the ADT algorithm is $\Delta\gamma = \gamma_{opt} - \gamma \leq \frac{\bar{r}T_o}{B}(2/\alpha_{min} - 1)$.

□

Figures 3.3(a) through 3.3(d) show the relationships among $\Delta\gamma$, $\bar{r}$, α, and T_o according to Theorem 1. Each figure plots $\Delta\gamma$ against $\bar{r}$ for $\alpha = 0.10, 0.20, 0.30, 0.40$, and 0.50 using a fixed value of T_o. The linear relationship of $\Delta\gamma$ with $\bar{r}$ is clear in the figures. It is also clear that decreasing α increases $\Delta\gamma$. The sequence of figures shows that $\Delta\gamma$ increases with increasing wake-up time T_o. Changing the value of α when T_o is small has little impact on $\Delta\gamma$, but when T_o and the average bit rate of the video streams, $\bar{r}$, are both large, the impact of changes in the value of α on $\Delta\gamma$ can be substantial. Similarly, increasing the value of T_o has little impact when α is small, but the impact can be quite large when α and $\bar{r}$ are both large. Therefore, if energy saving is critical, and both T_o and the average bit rate of the video streams are large, the operator can use α as a control parameter to adjust the system for more energy saving or a finer control over bandwidth which results in a better dropped frame rate.

Next, we show that our proposed algorithm is computationally efficient in the following theorem.

Theorem 2. *The ADT scheduling algorithm runs in time $O(NS)$, where N is the total number of control points and S is the number of video streams.*



(a) $T_o = 10$ms (b) $T_o = 25$ms

(c) $T_o = 50$ms (d) $T_o = 100$ms

Figure 3.3: Maximum possible gap ($\Delta\gamma$) for T_o values of 10ms, 25ms, 50ms, and 100ms.

Proof. We showed in the proof of Theorem 1 that the number of control points for video stream s is at most $u_s = r_s T/(\alpha B)$. We are assuming that the length of each video stream is at least αB, so each video stream has at least one control point. Let $N = \sum_{s=1}^{S} u_s$ be the total number of control points. At each control point we select the stream with the smallest deadline and schedule a burst for it. Using a priority queue, the stream s with the minimum deadline can be chosen in O(1) time. Adding a new deadline and control point to the priority queue for stream s takes time O($\log S$). Therefore, this process takes time O($N \log S$). The cost of scheduling a burst depends on the number of frames between two control points. Since the number of bytes in a burst cannot exceed the buffer size B, the number of frames that have to be considered is at most SB. Since B is a constant, the cost to schedule all bursts is O(NS) and the time complexity of the scheduling is O($N \log S + NS$) = O(NS). Also the number of times that the scheduling process will be run in a scheduling window will be constant, because there are a constant number of α values that we choose from through binary search. Therefore, the total time complexity of the algorithm is O(NS). □

We note that our algorithm is linear in terms of S (number of streams) and N (number of control points). Thus, our algorithm can easily run in real time. We have actually implemented our algorithm in a mobile video streaming testbed. The algorithm indeed runs in real time on a regular personal computer.

Next, we address the correctness of the proposed algorithm in the following theorem.

Theorem 3. *The ADT algorithm produces feasible burst schedules with no two bursts overlapping in time and it does not result in buffer overflow and underflow instances.*

Proof. First, we show that there always exists a value of α that results in no buffer underflow instances. In the ADT algorithm, there are time intervals for which we are scheduling bursts for video streams, and there will be slack times when no burst is being scheduled (Fig 3.4). The slack time could happen for two reasons. The first is that all receiver buffers have been completely filled, so no burst will be scheduled for them before their next control point. Between two consecutive control point αB Kb of data will be played out, so the buffer level of the receivers will be at least $(1-\alpha)B$ during the slack time. The second situation that we might have slack time is at the end of a scheduling window when all video data belonging to that scheduling window is scheduled and there is no data available. Since we assume that we have a small lookahead window and we can have access to data of as large as a buffer size from the next scheduling window, this case would never happen and we will not run

out of data at the end of a scheduling window. Therefore, in our algorithm, a slack time period will only be an instance of the first case we discussed above. Based on this, after a slack time and at the beginning of a duration of consecutive bursts, the buffer level for all receivers will be at least $(1-\alpha)B$ Kb. Now we prove the theorem by contradiction. We assume that there is a case for which there is no value of α that could avoid buffer underflow. We assume stream s to be the first video stream that will have buffer underflow in this case. Since, during the slack time, all buffer levels are at least $(1-\alpha)B$ Kb, buffer underflow for stream s happens at a time interval when we have consecutive scheduled bursts. We let t_b denote the beginning time of this consecutive bursts and t_u denote the time when buffer underflow occurs. We represent the time of the last control point before underflow time, t_u, by t_c. Since buffer underflow has occurred for stream s, this stream was not selected at the time t_c for burst scheduling. We represent the selected stream at time t_c by s'. Hence, the deadline for stream s' is before deadline for stream s, which is t_u. Since there is no control point between time t_c and time t_u, the buffer level of both streams at the time t_c is less than αB Kb. Now, if we denote the sum of the buffer levels of all video streams at time t by e_t, we will have $e_{t_c} < (S-2)B + 2\alpha B$, and $e_{t_b} \geq S(1-\alpha)B$. On the other hand, we have $e_{t_c} = e_{t_b} + R(t_c - t_b) - \sum_{s=1}^{S} r_s(t_c - t_b)$. Since $R \geq \sum_{s=1}^{S} r_s$, we will have $e_{t_c} \geq e_{t_b}$. Therefore we can say $(S-2)B + 2\alpha B > S(1-\alpha)B$. From this, we conclude that $\alpha > 2/(S+2)$. As a result, any value of $\alpha \leq 2/(S+2)$ will avoid buffer underflow which is contradicting the assumption that no value of α exists that could avoid buffer underflow. Also, at each scheduling time, we know the buffer level for stream s by subtracting the size of p_s frames that have been played out at the receiver from the total size of m_s frames that have been scheduled for the stream s. Based on this, we know the remaining capacity of the buffer and therefore, line 6 of the algorithm (Figure 3.2) guarantees that no buffer overflow occurs at the receiver buffers. Finally, we show that no two bursts overlap in time in the produced schedules. Each new burst is created starting at the current scheduling time $t_{schedule}$ in line 6 of the algorithm (Figure 3.2). After scheduling a burst in line 6, $t_{schedule}$ is updated in line 7 and moved forwards as much as the length of scheduled burst in line 6. Other than line 7, there is only one place where $t_{schedule}$ is updated in the algorithm and that is line 14. In line 14, $t_{schedule}$ is reset to the start time and the scheduling is started from scratch. Therefore, the ADT algorithm produces a feasible burst schedule for the base station. □

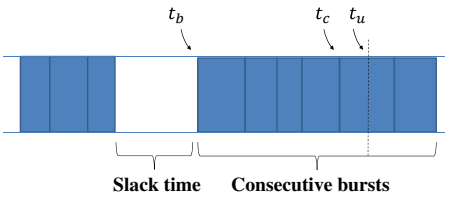


Figure 3.4: Scheduled bursts are free of buffer underflow instances

Chapter 4

Evaluation

In this chapter, we rigorously analyze the performance of the proposed algorithm and compare its performance to recent algorithms. We start by describing the setup of our mobile video testbed. Then, we present detailed results for different performance metrics.

4.1 Testbed and Setup

The testbed that we used to evaluate our algorithm, shown in Figure 4.1, consists of a base station, mobile receivers, and data analyzers. The base station includes an RF signal modulator which produces DVB-H standard-compliant signals. The signals are amplified to around 0 dB before transmission through a low-cost antenna to provide coverage to approximately 20m for cellular phones. The base station has a multi-threaded design that enables the broadcast of multiple video streams in real time on a low-end server.

The mobile receivers in our testbed are Nokia N96 cell phones that have DVB-H signal receivers and video players. Two DVB-H analyzers are included in the testbed system. The first one, a DiviCatch RF T/H tester [10], has a graphical interface for monitoring detailed information about the received signals, including burst schedules and burst jitters. The second analyzer, a dvbSAM [11], is used to access and analyze received data at the byte level to monitor the correctness of the received content.

We implemented our ADT algorithm, the SMS algorithm [24], and the Nokia Mobile Broadcast Solution (MBS) [1, 34], in the testbed and integrated them with the IP encapsulator of the transmitter. We set the overhead T_o to 100 msec and the modulator to a QPSK (Quadrature Phase Shift Keying) scheme and a 5 MHz radio channel. According to DVB-H

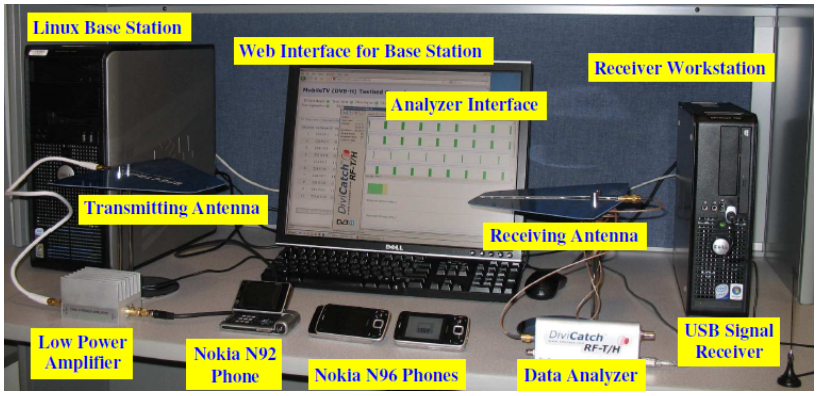


Figure 4.1: The mobile video streaming testbed used in the evaluation.

Bit rate (Kbps)	Description
83 – 121	8 video streams, documentary, movie, news
240 – 260	3 video streams, demo video and advertisements
400 – 600	6 video streams, movie, sport, news

Table 4.1: Video streams used in the experiments

standard documents, this gives us net bandwidth of 5.18 Mbps. We fixed the maximum receiver buffer size B at 4 Mb (500 KB). We prepared a test set of 17 diverse VBR video streams to evaluate the algorithms. The different content of the streams (TV commercials, sports, action movies, documentaries) provided a wide range of video characteristics with average bit rates ranging from 25 kbps to 600 kbps. The average bit rate of the video streams is 260 Kbps. Each video stream was played at 30 fps and had length of 566 sec. For more information about video streams you can refer to table 4.1. We transmitted the 17 VBR video streams concurrently to the receivers and we collect detailed statistics from the analyzers. Each experiment was repeated for each of the three algorithms (ADT, SMS, and MBS). In addition, for our ADT algorithm, we repeated the experiments with several values of the buffer adaptation parameter α.

4.2 Results for Dropped Video Frames

Dropped frames are frames that are received at the mobile receivers after their decoding deadlines or not received at all. The number of dropped frames is an important quality of service metric as it impacts the visual quality and smoothness of the received videos.

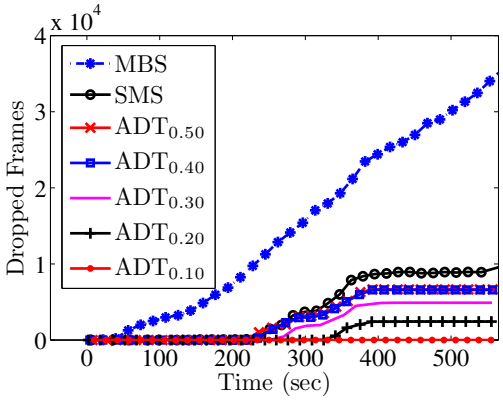
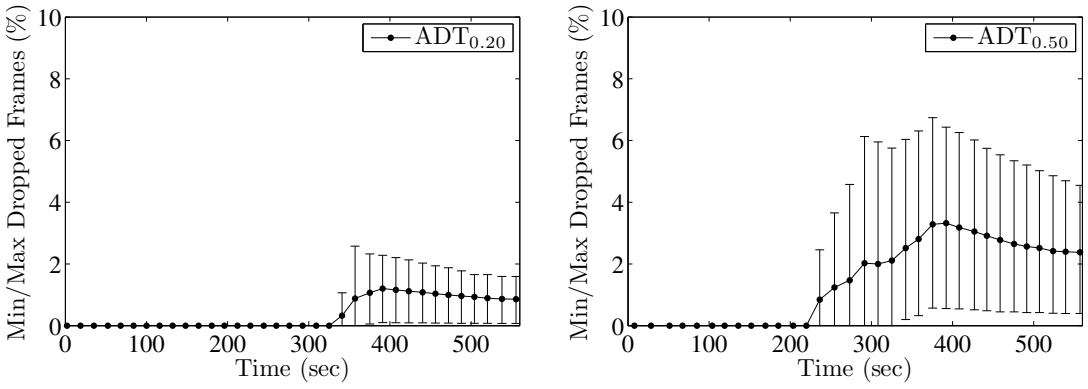


Figure 4.2: Total number of dropped video frames.



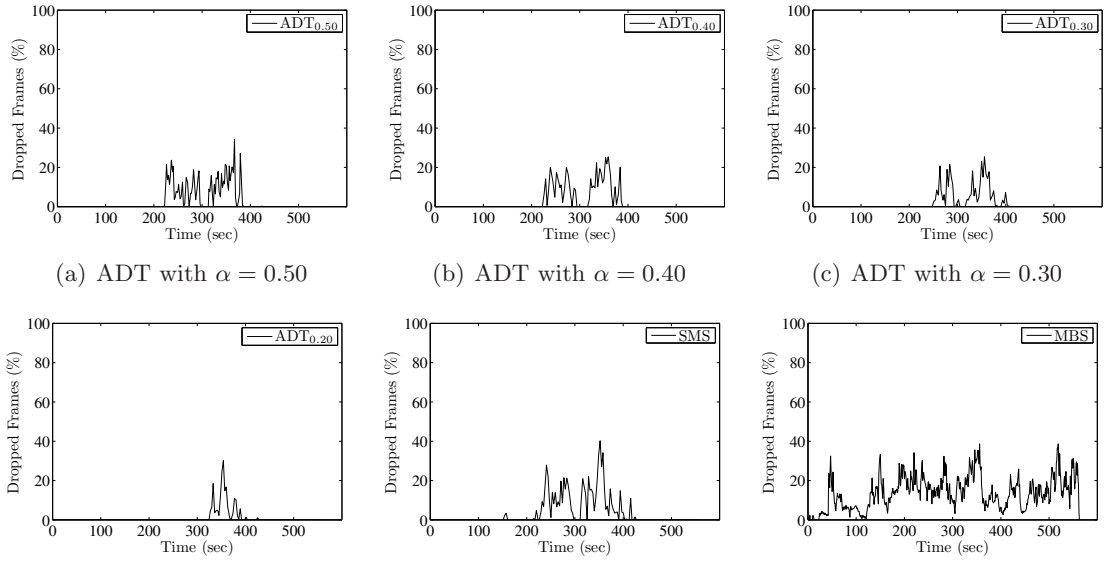
(a) $\alpha = 0.50$ (b) $\alpha = 0.20$

Figure 4.3: Minimum and maximum number of dropped video frames.

Figure 4.2 shows the cumulative total over all video streams of the numbers of dropped frames for ADT with fixed values of α ranging from 0.10 to 0.50, and for SMS and MBS. The figure clearly shows that our ADT consistently drops significantly fewer frames than the SMS and MBS algorithms. The figure also shows the effect of decreasing the value of α for our ADT algorithm. The total number of dropped frames decreases from 6630 with $\alpha = 0.50$ to 2074 with $\alpha = 0.20$. No frames are dropped when α is reduced to 0.10. On the other hand, the SMS algorithm is significantly worse with 9605 dropped frames. The results for MBS in Figure 4.2 were obtained by running the algorithm for each video stream with different assigned bit rates ranging from 0.25 times the average bit rate to 4 times the average bit rate of the video stream and then choosing the best result for each video stream. Even in this total of best cases, the number of dropped frames is more than 34000. In practice, an operator heuristically chooses the assigned bit rates for video streams, so the results in practice likely will be worse.

We counted the number of dropped frames for each video stream to check whether the ADT algorithm improves the quality of some video streams at the expense of others. Two samples of our results are shown in Figure 4.3; others are similar for different values of α. The curves in the figure show the average over all streams of the percentage of dropped frames; each point on the curves is the average percentage of frames transmitted to that point in time that were dropped. The bars show the ranges over all video streams. As shown in the figure, the difference between the maximum and minimum dropped frame percentages at the end of the transmission period is small. Therefore, the ADT algorithm does not sacrifice the quality of service for some streams to achieve good aggregate results.

We further analyze the patterns of dropped video frames for each algorithm in more detail. We plot the total number of dropped frames during each 1 sec interval across all video streams. Our results are shown in Figure 4.4. For the ADT algorithm with $\alpha = 0.50$, frames were dropped mostly during a 150 sec period (between 220 and 370 sec) because several video streams have high bit rate during this period. Reducing α to 0.20 permits finer control over the distribution of bandwidth among the video streams and much fewer frames were dropped as shown in Figure 4.4(d). Further reducing α to 0.10 eliminated all dropped frames as we have already seen in Figure 4.2. The SMS algorithm on the other hand dropped up to 42% of the frames during the period in which the aggregate bit rate of all streams is high. Using the MBS algorithm results in dropping frames in a very wide time range. The results from these experiments confirm the performance benefits that our



(a) ADT with $\alpha = 0.50$ (b) ADT with $\alpha = 0.40$ (c) ADT with $\alpha = 0.30$

(d) ADT with $\alpha = 0.20$ (e) SMS (f) MBS

Figure 4.4: Dropped video frames over 1 sec periods.

ADT algorithm achieves by dynamically adapting to the changing bit rate nature of VBR video streams by controlling the level of receivers' buffers through the parameter α.

We present the aggregate of average bit rate of all video streams over time in Fig. 4.5. We also show the aggregate of instantaneous bit rate of video streams in Fig. 4.6. As you can see, although the aggregate of average bit rate over time does not exceed bandwidth capacity, the aggregate of instantaneous bit rate of streams exceeds available bandwidth. We can see the effect of such changes in the average buffer level of the receivers in Fig. 4.7. We can see that when the instantaneous bit rate of video streams exceeds available bandwidth, the average buffer level of video streams will decrease. This is due to the fact that the size of played out data exceeds transmitted data. But when the available bandwidth exceeds the aggregate of instantaneous bit rate, the average buffer level of video receivers will increase again.

4.3 Results for Energy Saving

We compute the average energy saving γ achieved across all video streams, which represents the average amount of time that the wireless interface is in off mode. The results are shown in Figure 4.8. The figure shows that the average saving resulted from the ADT algorithm

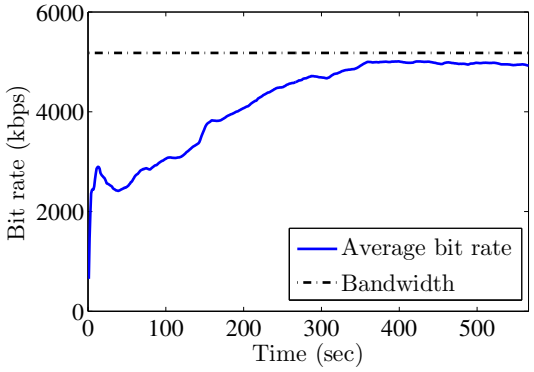


Figure 4.5: Aggregate of average bit rate of all video streams over time.

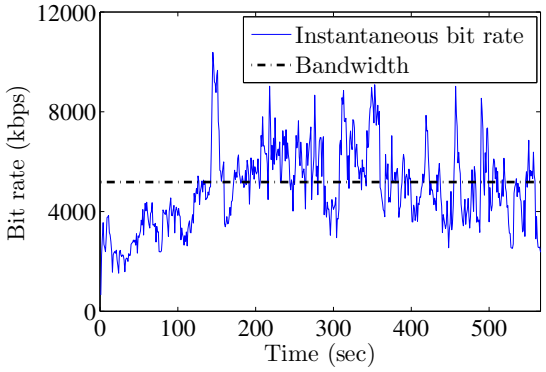


Figure 4.6: Aggregate of instantaneous bit rate of all video streams.

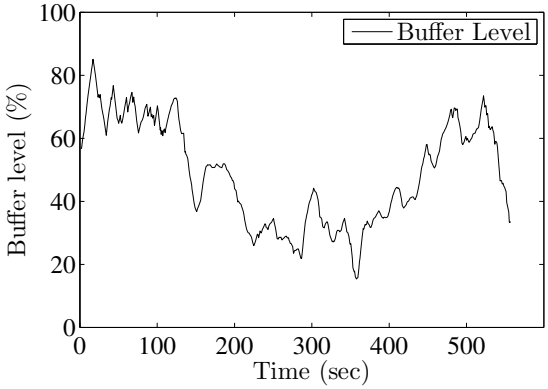


Figure 4.7: Average buffer level of receivers.

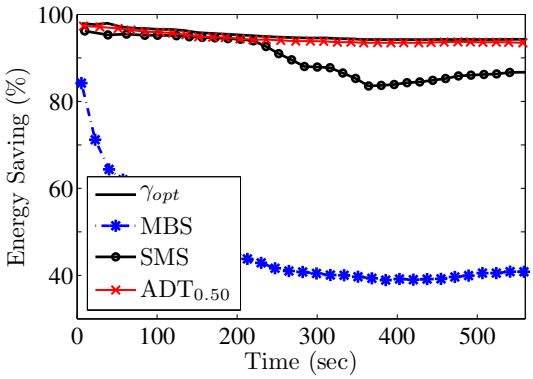


Figure 4.8: Average energy saving.

when $\alpha = 0.5$ is very close to the optimal energy saving, and it is about 6.74% more than the average energy saving achieved by the SMS algorithm. Also, our experiments show that the energy saving achieved by ADT is considerably higher than the MBS algorithm, which achieves an average energy saving of up to 41.14%.

The impact of changing α on the energy saving achieved by the ADT algorithm is shown in Figure 4.9. The average over all video streams of the energy saving is 89.52% for the ADT algorithm when $\alpha = 0.10$ which is better than the SMS algorithm. Increasing α to 0.50 increases the energy saving to 93.47%. The small improvement of 3.95% in energy saving is non-trivial but it might not be large enough in many practical situations to offset the advantage of minimizing the number of dropped frames by setting $\alpha = 0.10$.

Next, we compare the energy saving resulted from the ADT algorithm against the upper and lower bounds resulting from Theorem 1. Figure 4.10 shows the results for α values of 0.5, 0.3, and 0.1 is shown. It is important to note that the upper bound in the figure is not the optimal energy saving, but it is a conservative upper bound on the optimal energy saving which may not be achievable. According to the figure, the gap between our algorithm and the conservative upper bound is less than 5% for $\alpha = 0.5$.

Finally, we measured the energy saving for the receivers of each individual stream to check whether the ADT algorithm unfairly saves more energy for some receivers at the expense of others. A sample of our results is shown in Figure 4.11. The figure confirms that ADT does not sacrifice energy saving in some streams to achieve good average energy saving.

In summary, the results in this section show that the proposed algorithm achieves higher energy saving than the previous algorithms, the saving is uniform across all mobile receivers, and the saving is very close to the optimal energy saving.

4.4 Impact of Changing α

The results in the previous two subsections indicated that changing the value of the buffer control parameter α impacts the number of dropped frames and the energy saving achieved by mobile devices. In this subsection, we analyze this impact for different values of α. We change the value of α from 0.10 to 0.50 for our ADT algorithm. For each value of α, we ran the experiment of transmitting all video streams and measured the average energy saving as well as the percentage of video frames that arrived on time. We plot the results

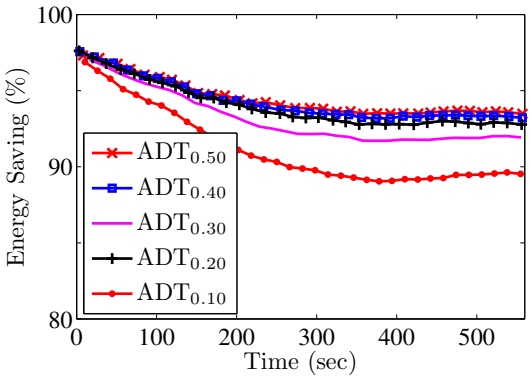
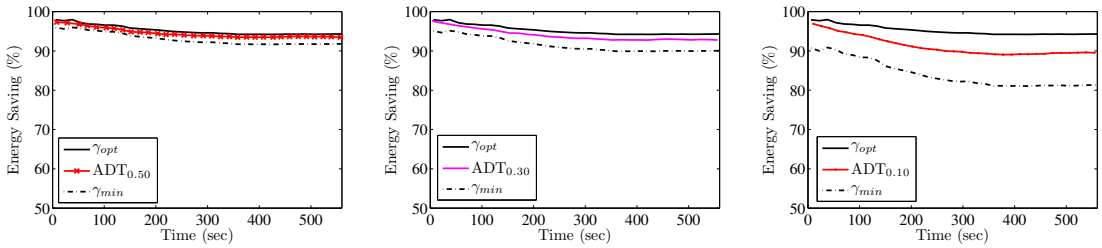


Figure 4.9: Average energy saving with different values of α.



(a) $\alpha = 0.50$ (b) $\alpha = 0.30$ (c) $\alpha = 0.10$

Figure 4.10: Upper and lower bounds for energy saving.

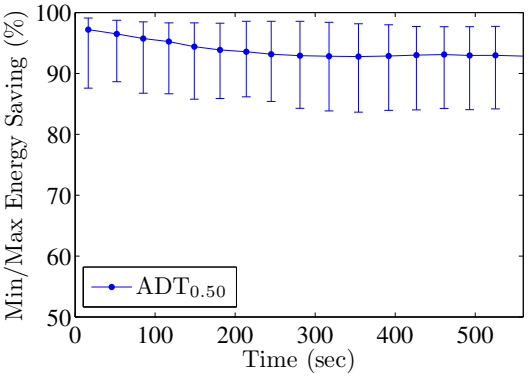


Figure 4.11: Min and max energy saving.

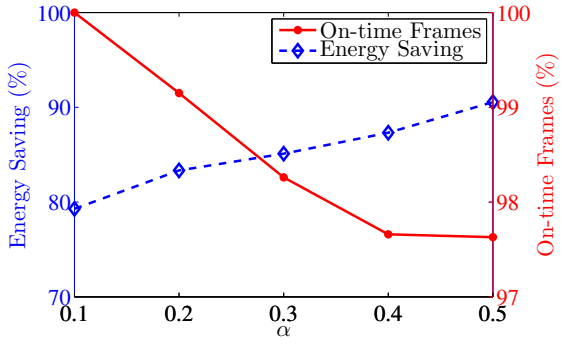


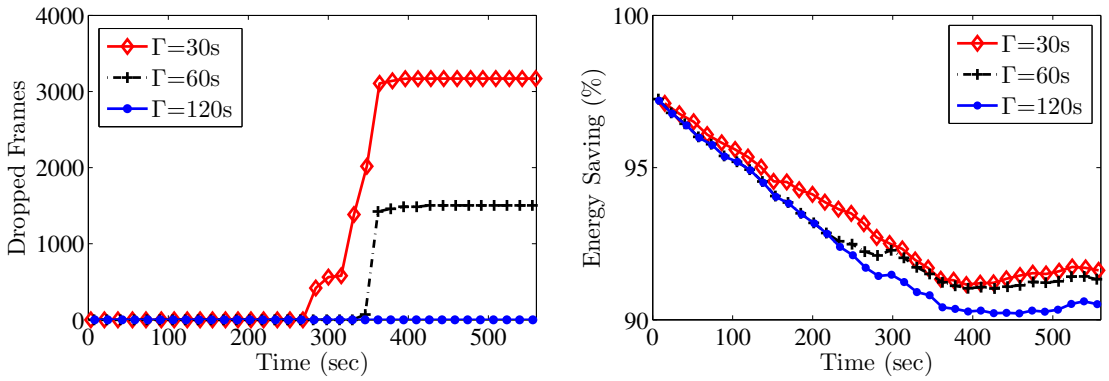
Figure 4.12: The impact of α on energy saving and number of dropped video frames.

in Figure 4.12. Notice that the figure has two y-axes. The results show that α exposes a tradeoff between the energy saving and the percentage of frames that arrive on time. Small α values allow finer control of the wireless channel and thus increase the chance of transmitting video frames on time, but small values of α result in more bursts being used to transmit the same amount of video data, and each burst incurs an overhead of T_o. Thus, small α values result in smaller energy saving. Based on this, the operators can use larger values of α for lower bit rate video streams and smaller values of α to have finer control over the bandwidth of high bit rate video streams. It is important to note that even with the smallest value for α, i.e., $\alpha = 0.10$, our ADT algorithm still achieves energy saving as high as the state-of-the-art SMS algorithm.

4.5 Dynamic Adjustment of α

The previous subsection shows the existence of a tradeoff between energy saving and the number of on-time video frames, which is controlled by α. Our ADT algorithm is capable of adaptively updating the value of α at run time based on bandwidth limitations and changes in the bitrates of the video streams to give a good balance between the number of dropped frames and energy saving. This is illustrated in the pseudo code of the ADT algorithm in Figure 3.2, where the value of α is dynamically adjusted. The ADT algorithm tries to maintain the largest value of α that achieves the minimum number of dropped frames because larger values of α allow greater energy saving.

We conducted an experiment to show the merits of dynamically adjusting the value of α in the range of 0.1 to 0.5, instead of fixing it as in the previous subsections. We ran our ADT algorithm and allowed it to change α at run time. We chose different values for the scheduling window Γ: 30, 60, 120 sec. The results are given in Figure 4.13. When the scheduling window is 30 sec, the energy saving converges to 92% (Figure 4.13(b)) which is approximately the same as with a fixed value of $\alpha = 0.3$, but the total number of dropped frames for each video stream is 3162 (Figure 4.13(a)) which is much better than the 4896 dropped frames with fixed $\alpha = 0.3$. When the scheduling window is 60 sec, the energy saving is 91.6% which is similar to the energy saving results with a fixed value of $\alpha = 0.2$ but the total number of dropped frames for each streams is 1496 which is much better than 2414 with fixed $\alpha = 0.2$. With a scheduling window of 120 sec, no frames are dropped and the energy saving converges to 90.51% which is better than energy saving of 89.52% results



(a) Dropped frames (b) Energy saving

Figure 4.13: The impact of dynamically choosing α.

with a fixed value of $\alpha = 0.1$ which also resulted in 0 dropped frames.

In summary, the results of this experiment show that dynamically choosing the parameter α as done by our algorithm achieves both good energy saving and small numbers of dropped video frames by dynamically using smaller values of α when we need finer control over bandwidth, and larger values of α when we do not have bandwidth limitations. And, more importantly, network operators do not need to heuristically set any parameters for our ADT algorithm.

Chapter 5

Conclusions and Future Work

In this chapter, we summarize our contributions in this thesis and then we highlight the directions in which this work could be extended.

5.1 Conclusions

We have presented a new algorithm for transmitting multiple VBR video streams to energy-constrained mobile devices. The algorithm is to be used by base stations in wide-area wireless networks that offer multimedia services in broadcast/multicast modes such as the MBMS (Multimedia Broadcast Multicast Service) of 3G/4G cellular networks, WiMAX networks, and DVB-H (Digital Video Broadcast–Handheld) networks. One of the novel aspects of the proposed algorithm is its ability to dynamically adjust the level of receivers' buffers according to the bit rates of the video streams being received by each receiver. This is done through dynamically controlling the allocation of wireless bandwidth to video streams. We show that the proposed algorithm computes feasible schedules whenever they exist. We analytically computed the gap between the energy saving resulting from our algorithm and the optimal energy saving, and we showed that this gap is small. We presented a proof-of-concept implementation of the proposed algorithm in a mobile video streaming testbed. We compared the new algorithm against other algorithms proposed in the literature and used in practice. We conducted extensive empirical analysis using a number of VBR video streams with diverse visual characteristics and bit rates. Our results show that the proposed algorithm yields high energy saving for mobile receivers and reduces the number of video frames that miss their deadlines. The results also demonstrate that the proposed algorithm

outperforms the current state-of-the-art algorithms.

5.2 Future Work

- We can extend the quality of service metrics considered in our work. In video encoding standards, different frames could have varying effects on the quality of decoded video. We can extend our formulation to differentiate among video frames and define more complicated quality of service metrics to consider video quality.

- In this thesis, we proposed a general algorithm that is not dependent on any specific standard. But video streaming standards such as MediaFLO [18] and WiMax [45] use different multiplexing formats and deploy various data encapsulation techniques. Considering characteristics of each standard while developing a solution could result in more efficient multiplexing algorithms for them.

- In this thesis, we developed our algorithm for the case where no feedback channel for the base station exists. However, video streaming applications could be interactive and the base station could receive feedback from mobile devices. Interactivity of the applications will impose many quality of service requirements for the video streaming services that could be a topic of research.

- This work could be extended to support temporal and quality scalability for scalable video streams. Considering the opportunities provided through temporal and quality scalability of scalable video streams could result in better solutions for multiplexing scalable video.

- Although transmitting video data in large bursts results in more energy saving for the receivers, it could increase channel switching delay. Considering channel switching delay as an additional performance metric could be an interesting extension for this work.

Bibliography

[1] Private communication with Nokia's engineers managing mobile TV base stations.

[2] Multimedia Broadcast/Multicast Service (MBMS) in GERAN; Stage 2 (Release 6). Third Generation Partnership Project (3GPP) Standard TS 43.246 Ver. 9.0.0, 2009.

[3] Multimedia Broadcast/Multicast Service (MBMS); stage 1: User services. Third Generation Partnership Project (3GPP) Standard TS 22.246 Ver. 9.0.0, 2009.

[4] ATSC mobile DTV standard, 2009. `http://www.openmobilevideo.com/about-mobile-dtv/standards/`.

[5] AT&T sells wireless spectrum in southeast to Clearwire corporation. `http://www.att.com/gen/press-room?pid=4800&cdvn=news&newsarticleid=23428`.

[6] P. Camarda, G. Tommaso, and D. Striccoli. A smoothing algorithm for time slicing DVB-H video transmission with bandwidth constraints. In *Proc. of ACM International Mobile Multimedia Communications Conference (MobiMedia'06)*, Alghero, Italy, September 2006.

[7] Craig wireless to sell canadian spectrum for $80m, 2010. `http://www.cbc.ca/fp/story/2010/03/26/2729450.html`.

[8] M. Chari, F. Ling, A. Mantravadi, R. Krishnamoorthi, R. Vijayan, G. Walker, and R. Chandhok. FLO physical layer: An overview. *IEEE Transactions on Broadcasting*, 53(1):145–160, March 2007.

[9] Wu chi Feng and Stuart Sechrest. Critical bandwidth allocation for the delivery of compressed video. *Computer Communications*, 18(10):709 – 717, 1995. System Suport for Multimedia Computing.

[10] Divi Catch RF-T/H transport stream analyzer, 2008. `http://www.enensys.com/`.

[11] dvbSAM DVB-H solution for analysis, monitoring, and measurement, 2008. `http://www.decontis.com/`.

[12] Digital Video Broadcasting (DVB); framing structure, channel coding and modulation for digital terrestrial television. European Telecommunications Standards Institute (ETSI) Standard EN 300 744 Ver. 1.5.1, June 2004.

[13] Digital Video Broadcasting (DVB); transmission system for handheld terminals (DVB-H). European Telecommunications Standards Institute (ETSI) Standard EN 302 304 Ver. 1.1.1, November 2004.

[14] Digital Video Broadcasting (DVB); DVB-H implementation guidelines. European Telecommunications Standards Institute (ETSI) Standard EN 102 377 Ver. 1.3.1, May 2007.

[15] G. Faria, J. Henriksson, E. Stare, and P. Talmola. DVB-H: Digital broadcast services to handheld devices. *Proc. of the IEEE*, 94(1):194–209, January 2006.

[16] W. Feng and J. Rexford. Performance evaluation of smoothing algorithms for transmitting prerecorded variable-bit-rate video. *IEEE Transactions on Multimedia*, 1(3):302–312, September 1999.

[17] Wu-chi Feng, Farnam Jahanian, and Stuart Sechrest. An optimal bandwidth allocation strategy for the delivery of compressed prerecorded video. *Multimedia Syst.*, 5:297–309, September 1997.

[18] FLO technology overview, 2009. `http://www.mediaflo.com/news/pdf/tech_overview.pdf`.

[19] Global IPTV market analysis (2006–2010). `http://www.rncos.com/Report/IM063.htm`.

[20] Matthias Grossglauser, Srinivasan Keshav, and David N. C. Tse. Rcbr: a simple and efficient service for multiple time-scale traffic. *IEEE/ACM Trans. Netw.*, 5:741–755, December 1997.

[21] GSM Home Page. `http://www.gsmworld.com/`.

[22] Z. He and D. Wu. Linear rate control and optimum statistical multiplexing for H.264 video broadcast. *IEEE Transactions on Multimedia*, 10(7):1237–1249, November 2008.

[23] M. Hefeeda and C. Hsu. On burst transmission scheduling in mobile TV broadcast networks. *IEEE/ACM Transactions on Networking*, 18(2):610–623, April 2010.

[24] C. Hsu and M. Hefeeda. On statistical multiplexing of variable-bit-rate video streams in mobile systems. In *Proc. of ACM Multimedia'09*, pages 411–420, Beijing, China, October 2009.

[25] C. Hsu and M. Hefeeda. Time slicing in mobile TV broadcast networks with arbitrary channel bit rates. In *Proc. of IEEE INFOCOM'09*, pages 2231–2239, Rio de Janeiro, Brazil, April 2009.

[26] IEEE Standard. Local and metropolitan area networks Part 16: Air Interface for Broadband Wireless Access Systems Broadband Wireless Metropolitan Area Network. `http://standards.ieee.org/getieee802/download/802.16-2009.pdf`.

[27] M. Jacobs, J. Barbarien, S. Tondeur, R. Van de Walle, T. Paridaens, and P. Schelkens. Statistical multiplexing using SVC. In *Proc. of IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB'08)*, pages 1–6, Las Vegas, NV, March 2008.

[28] M. Kornfeld and G. May. DVB-H and IP Datacast – broadcast to handheld devices. *IEEE Transactions on Broadcasting*, 53(1):161–170, March 2007.

[29] Marwan Krunz and Herman Hughes. A traffic model for mpeg-coded VBR streams. In *In Proc. of the ACM SIGMETRICS/PERFORMANCE '95 Conference*, pages 47–55, 1995.

[30] H. Lai, J. Lee, and L. Chen. A monotonic-decreasing rate scheduler for variable-bit-rate video streaming. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(2):221–231, February 2005.

[31] T. Lakshman, A. Ortega, and A. Reibman. VBR video: Tradeoffs and potentials. *Proc. of the IEEE*, 86(5):952–973, May 1998.

[32] J. Lin, R. Chang, J. Ho, and F. Lai. FOS: A funnel-based approach for optimal online traffic smoothing of live video. *IEEE Transactions on Multimedia*, 8(5):996–1004, October 2006.

[33] Farid Molazem Tabrizi, Joseph Peters, and Mohamed Hefeeda. Adaptive transmission of variable-bit-rate video streams to mobile devices. In *NETWORKING 2011*, Lecture Notes in Computer Science. Springer-Verlag, 2011.

[34] Nokia mobile broadcast solution. `http://press.nokia.com/PR/200510/1018770_5.html`.

[35] S. Parkvall, E. Englund, M. Lundevall, and J. Torsner. Evolving 3G mobile systems: Broadband and broadcast services in WCDMA. *IEEE Communications Magazine*, 44(2):30–36, February 2006.

[36] Amy R. Reibman and Arthur W. Berger. Traffic descriptors for vbr video teleconferencing over atm networks. *IEEE/ACM Trans. Netw.*, 3:329–339, June 1995.

[37] M. Rezaei. Video streaming over DVB-H. In F. Luo, editor, *Mobile Multimedia Broadcasting Standards*, chapter 4, pages 109–131. Springer US, November 2009.

[38] M. Rezaei, I. Bouazizi, and M. Gabbouj. Joint video coding and statistical multiplexing for broadcasting over DVB-H channels. *IEEE Transactions on Multimedia*, 10(7):1455–1464, December 2008.

[39] J. Ribas-Corbera, P. Chou, and S. Regunathan. A generalized hypothetical reference decoder for H.264/AVC. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):674–687, July 2003.

[40] James D. Salehi, Shi-Li Zhang, Jim Kurose, and Don Towsley. Supporting stored video: reducing rate variability and end-to-end resource requirements through optimal smoothing. *IEEE/ACM Trans. Netw.*, 6:397–410, August 1998.

[41] S. Sen, J. Rexford, J. Dey, J. Kurose, and D. Towsley. Online smoothing of variable-bit-rate streaming video. *IEEE Transactions on Multimedia*, 2(1):37–48, March 2000.

[42] Farid Molazem Tabrizi, Cheng-Hsin Hsu, Mohamed Hefeeda, and Joseph G. Peters. Optimal scalable video multiplexing in mobile broadcast networks. In *Proceedings of the 3rd Workshop on Mobile Video Delivery (MoViD'10)*, pages 9–14. ACM, 2010.

[43] M. Tagliasacchi, G. Valenzise, and S. Tubaro. Minimum variance optimal rate allocation for multiplexed H.264/AVC bitstreams. *IEEE Transactions on Image Processing*, 17(7):1057–1143, July 2008.

[44] Patrick Thiran, Jean yves Le Boudec, and Frederic Worm. Network calculus applied to optimal multimedia smoothing. In *Proc. of IEEE INFOCOM'01*, pages 1474–1483, Anchorage, Alaska, April 2001.

[45] IEEE 802.16: Broadband Wireless Metropolitan Area Network, 2009. `http://standards.ieee.org/getieee802/802.16.html`.

[46] X. Yang, Y. Song, T. Owens, J. Cosmas, and T. Itagaki. Performance analysis of time slicing in DVB-H. In *Proc. of Joint IST Workshop on Mobile Future and Symposium on Trends in Communications (SympoTIC'04)*, pages 183–186, Bratislava, Slovakia, October 2004.

[47] Junbiao Zhang and Joseph Hui. Applying traffic smoothing techniques for quality of service control in vbr video transmissions. *Computer Communications*, 21(4):375 – 389, 1998. Quality of Services in Distributed Systems.