Optimal Scalable Video Multiplexing in Mobile Broadcast Networks

Farid Molazem Tabrizi
School of Computing Science
Simon Fraser University,

Cheng-Hsin Hsu
Deutsche Telekom Inc.
R&D Laboratories USA

Mohamed Hefeeda
School of Computing Science
Simon Fraser University,

Joseph G. Peters
School of Computing Science
Simon Fraser University,

ABSTRACT

We study the problem of broadcasting multiple variable bit rate scalable video streams from a base station to a large number of mobile receivers. Our objective is to maximize bandwidth utilization, energy saving and perceived quality of the transmitted videos. In practice, the aggregate bitrate of the video streams can be greater than the available bandwidth and then the receivers may experience playout glitches. We propose an algorithm to take advantage of opportunities provided by scalable video coding to improve performance when the bandwidth is limited. To achieve this, we provide a model for prioritizing video packets coded in medium grain scalability and a decision mechanism to drop quality layers of video frames when the bandwidth is limited. The result is increased bandwidth utilization and quality of the transmitted videos. Our experiments, performed on a real mobile TV testbed, show that our algorithm significantly out-performs current methods in terms of bandwidth utilization and also achieves near optimal energy saving and quality of the transmitted video streams.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless Communication*

General Terms

Design

Keywords

Mobile video broadcasting, scalable video coding, transmission scheduling, wireless video streaming

1. INTRODUCTION

Recent advances in mobile handheld devices have resulted in considerable interest in mobile video streaming in both the research community and commercial markets [7]. However, digital video streaming is still facing many challenges in terms of quality of service, energy consumption, and bandwidth limitations. The wireless medium is sensitive to errors, congestion, limited bandwidth, latency, packet loss, and jitter which reduce the expected quality of service [13]. Also, the limited battery life of mobile receivers should be considered when designing a video streaming method. Another very important consideration is the expense of wireless spectrum. For example, AT&T recently sold a 2.5 GHz spectrum to Clearwire Corporation for $300 million [3].

In this paper, we propose an algorithm to broadcast multiple variable bit rate (VBR) scalable video streams from a base station to a large number of mobile receivers. Our algorithm can be used in broadcasting networks such as DVB-H (Digital Video Broadcast-Handheld) [6], and ATSC M/H (Advanced Television Systems Committee-Mobile/Handheld) [2]. The primary goal of our algorithm is to maximize bandwidth utilization. The secondary optimization objective are to maximize energy saving and the perceived quality of the transmitted video.

We compare the results of our algorithm to the SMS algorithm proposed in [8] which uses burst scheduling of VBR streams to provide optimal bandwidth utilization and near optimal energy saving when the available wireless network bandwidth is not a limiting factor. In particular, SMS assumes that the aggregate bitrate of the video streams does not exceed the available bandwidth. Furthermore, SMS has no mechanism to consider the quality of the frames that it drops. Controlling the aggregate bitrate of video streams could be possible to some extent by using joint video coders [15] which consist of a joint rate allocator, decoders, and VBR coders to encode video streams and dynamically allocate bandwidth among them so that the network is not overloaded. But in practice, many broadcast networks are not equipped with these components due to the added expense and complexity. In these cases, the aggregate bitrate of the video streams could instantaneously exceed the network bandwidth. If this happens, then the SMS algorithm could unexpectedly drop video data resulting in buffer underflow and playout glitches.

The new multiplexing algorithm that we present in this paper leverages scalable video coding (SVC) to extend the SMS algorithm in order to handle bandwidth limitations. Our algorithm provides significantly better bandwidth utilization than SMS when the aggregate bitrate of the video streams exceeds the available bandwidth while providing near optimal video stream quality and energy saving. To achieve these performance improvements, we have developed a model of the quality of video streams and defined a

MoViD'10, October 25, 2010, Firenze, Italy.

weighting mechanism for video data. When the aggregate bitrate of full-quality video streams exceeds the available bandwidth, our algorithm drops video packets based on this weighting mechanism and achieves R-D (rate-distortion) optimized video broadcasting. Our results show that the average quality of the resulting video streams is close to the quality that the streams would have when there is no packet loss, and about 1.34 dB higher than the SMS algorithm. We also show that our multiplexing algorithm is quite energy efficient. In fact, its performance is very close to the maximum possible energy saving; on average 90% energy saving is achieved.

The rest of this paper is organized as follows. We discuss the recent SVC standard in Sec. 2. In Sec. 3, we survey related work in the literature. We describe the multiplexing problem that we are studying in Sec. 4, and we present our new algorithm in Sec. 5. In Sec. 6, we extensively evaluate our solution using experiments on a real mobile TV testbed. Sec. 7 concludes this paper.

2. SCALABLE VIDEO CODING

We propose to leverage the features of the Medium Grain Scalability (MGS) mode of the Scalable Video Coding (SVC) extension of H.264/AVC [17] in mobile broadcast networks. SVC is an interesting coding method for modern video streaming systems which are mostly based on RTP/IP and have receivers with different connection qualities, processing powers, and quality of service requirements. Three scalability modes in SVC are temporal, spatial, and quality scalability. We will concentrate on quality scalability in which a video stream with lower quality layers than the original video will have the same spatial-temporal resolution, but lower fidelity. There are three modes defined for quality layers in SVC: Fine Grain Scalability (FGS), Coarse Grain Scalability (CGS), and Medium Grain Scalability (MGS) [17]. In FGS, motion compensation is performed on the lowest quality layer (base layer), so the encoder and the decoder always use consistent reference pictures and the loss/truncation of enhancement layer packets does not lead to error drifting in motion compensation for subsequent pictures. The main problem with FGS is that coding is very inefficient due to the inferior quality of base layer reconstruction. In CGS, motion compensation is performed separately for each layer. This addresses the coding complexity problem of FGS, but any losses in the quality enhancement layer will accumulate over time and cause serious quality degradation. To address this problem, switching between quality layers in CGS is only possible at I-frames. The third mode, MGS, uses the concept of *key pictures* [14] and employs a flag for each picture indicating whether the base layer or the enhancement layer reconstruction of the reference picture is used for motion compensation. Hence, in MGS, motion prediction can still be performed in the enhancement layer, but the encoder's and decoder's motion compensation loops are guaranteed to be synchronized at the periodic key pictures. This provides a trade off between the drifting issue and coding efficiency in MGS and provides the ability to switch between quality layers at any frame. In this paper, we assume that video streams are coded with MGS quality layers. Our algorithm takes advantage of the flexibility of MGS coding to maximize bandwidth utilization while providing near optimal video stream quality and energy saving.

3. RELATED WORK

A number of methods have been proposed to improve quality, energy saving, and frame loss in video streams over mobile broadcast networks. Some of these methods use smoothing algorithms to reduce the burstiness of VBR streams, which results in less frame loss in cases of bandwidth limitations. An online smoothing algorithm is provided in [18] and it is shown that delaying a video transmission by a constant amount of time can reduce the bandwidth required. Camarda et al. [4] design a smoothing algorithm for mobile TV, which takes into account the receiver buffer level, the available bandwidth, and the burst size. These studies [18, 4] do not consider multiple video streams or how the available bandwidth should be distributed among them to increase bandwidth utilization, energy saving, and average video quality. Thus, they do not provide an efficient solution to our problem.

Some authors have considered the characteristics of SVC streams to improve the quality of video streams. In [16], the potential benefits of using SVC in mobile networks are described and several uses in mobile broadcast, multicast, and unicast networks that could benefit from SVC are discussed. Strategies for dynamic sharing of radio links in a multi-user streaming network is proposed in [10]. The authors combine SVC with radio link buffer management strategies and try to maximize the video quality of the transmitted video streams by dropping the packets in a way that does not violate temporal scalability. However, they do not consider bandwidth utilization or energy saving, and the buffers of the receivers are unlimited.

The multiplexing techniques being used in practical mobile broadcast networks are not very efficient. For example, the Nokia Mobile Broadcast Solution (MBS) [11] requires the network operators to choose a time interval ΔT and a burst size b_s and bitrate r_s for each video stream s [1]. The value of ΔT is set to B/r_S where B is the buffer size of the receivers and r_S is the maximum of the bitrates assigned to the streams. During each time interval of length ΔT, each video stream is assigned bandwidth proportional to its bitrate. The network operator heuristically chooses r_s for each video stream. This can be a time-consuming process because higher values of r_s could result in buffer overflow and lower values of r_s could result in buffer underflow at the receivers. In this paper, we will compare our solution with the MBS method. Our results show that our algorithm significantly outperforms this method.

Our previous work addresses the multiplexing problem in mobile broadcast networks for constant bit rate (CBR) streams [9], and partially solves the same problem for VBR streams [8] under the assumption that the aggregate bitrate of video streams never exceeds the available bandwidth. In contrast, the present work eliminates the bitrate assumption and studies a more realistic scenario of broadcasting VBR streams.

4. PROBLEM STATEMENT

We study the problem of broadcasting S scalable video streams from a base station to a large number of mobile receivers over a wireless medium with bandwidth R kbps. Each video stream s, $1 \leq s \leq S$, has a base layer, Q_s MGS layers, I frames, and is coded at F frames-per-second. We use $l_{i,s,k}$ to denote the size of layer k of frame i of video stream s. Layer $k = 0$ is the base layer. We define on-time frames to be frames that are received at the decoders before their decoding deadlines. Our bandwidth utilization metric, called goodput and denoted σ, is the fraction of data transmitted by the base station over the wireless medium that is on-time. We define the energy saving of video stream s, denoted γ_s, to be the fraction of the total transmission time that the receivers can put their wireless interfaces to sleep. Our energy saving metric is the average energy saving of all streams: $\gamma = (\sum_{s=1}^{S} \gamma_s)/S$. We use ϕ to denote the average quality of all transmitted frames of all video streams. Different metrics can be used to measure the quality of a frame. We will use Peak Signal-to-Noise Ratio (PSNR) as the quality metric in our experiments, but Structural Similarity

(SSIM), Video Quality Metric (VQM), or any other quality metric can be substituted in our formulation. Now, based on these definitions, we can state the problem that we study in this paper: given S scalable video streams, find a burst schedule that maximizes goodput σ, with high energy saving γ and high quality ϕ.

We formulate the problem as follows. Let n_s be the number of scheduled bursts for video stream s where $1 \leq s \leq S$ and let f_k^s sec and b_k^s kb ($1 \leq k \leq n_s$) be the start time and the burst size of burst k in video stream s. To receive a burst, the receiver circuits must be wakened up a short time before the burst is received to lock onto the frequency and synchronize to the symbols. This overhead time, T_0, can vary from 50 to 250 msec [5]. Thus, for each burst k, the receiving circuits should be active in the period $[f_k^s - T_0, f_k^s + b_k^s/R)$. Finally, we let B denote the receiver buffer size and define c_k^s, the buffer level of mobile receivers before receiving burst k of video stream s as follows:

$$c_k^s = max(0, \sum_{j=1}^{k-1} b_j^s - \sum_{i=1}^{v} \sum_{q=0}^{u_i^s} l_{i,s,q}). \quad (1)$$

In Eq. (1), u_i^s is the number of MGS layers in frame i of video stream s and v is the maximum positive integer such that $v/F \leq f_k^s$. Thus c_k^s is the difference between the amount of data that has been received and the amount that the receiver has consumed. We use g_k^s and h_k^s to denote the first and last frames, respectively, of burst k of video stream s. We define $\lambda_{i,s,q}$ to be the quality improvement resulting from adding quality layer q to frame i of stream s.

Using these definitions, we formally describe the multiplexing problem with Eq. (2) below. In this formulation, our primary objective, expressed in Eq. (2a), is to maximize the goodput, the fraction of transmitted data that is on time. Eqs. (2b) and (2c) are our secondary optimization criteria. Eq. (2b) expresses the average energy saving for the mobile receivers as the average fraction of total time that the receivers can put their wireless interfaces to sleep. Eq. (2c) is the average quality of the video streams per kilo-byte of data and is based on the contribution of each MGS layer of each video stream to the quality of that stream. Eq. (2d) guarantees that the bursts of every pair s and $\overline{s}$ do not overlap. Eqs. (2e) and (2f) guarantee that we do not have any instances of overflow or underflow, respectively, at the receivers. Eq. (2g) relates the burst sizes and the numbers of MGS layers and is needed in Eq. (2c) to ensure that the sum of the transmitted MGS packets in each burst does not exceed the burst size.

$$\text{Pri:} \quad \max\ \sigma = \frac{\sum_{s=1}^{S} \sum_{j=1}^{n_s} b_j^s / R}{I/F}, \quad (2a)$$

$$\text{Sec:} \quad \max\ \gamma = 1 - \frac{\sum_{s=1}^{S} \sum_{k=1}^{n_s} (T_0 + b_k^s/R)}{I/F} / S, \quad (2b)$$

$$\text{Sec:} \quad \max\ \phi = \frac{\sum_{s=1}^{S} \sum_{k=1}^{n_s} \sum_{i=g_k^s}^{h_k^s} \sum_{q=1}^{u_i^s} \lambda_{i,s,q}}{\sum_{k=1}^{n_s} b_k^s}, \quad (2c)$$

$$\text{s.t.} \quad [f_k^s, f_k^s + \tfrac{b_k^s}{R}) \cap [f_{\overline{k}}^{\overline{s}}, f_{\overline{k}}^{\overline{s}} + \tfrac{b_{\overline{k}}^{\overline{s}}}{R}) = \varnothing, \quad (2d)$$

$$c_k^s + b_k^s - \sum_{f_k^s \leq j/F < f_k^s + b_k^s/R} \sum_{q=0}^{u_i^s} l_{j,s,q} \leq B, \quad (2e)$$

$$c_k^s \geq 0, \quad (2f)$$

$$b_k^s \geq \sum_{i=g_s^k}^{h_s^k} \sum_{q=0}^{u_i^s} l_{i,s,q}, \quad (2g)$$

$$\forall\, 1 \leq s \neq \overline{s} \leq S,\ 1 \leq k \leq n_s,\ 1 \leq \overline{k} \leq n_{\overline{s}}.$$

5. PROBLEM SOLUTION

In order to solve this problem, we divide the receiver buffer of size B into two buffers of the size $B/2$. We also divide the sending time of each video stream s into p_s disjoint time windows. These time windows for different video streams are completely independent of each other. During each time window p of video stream s, at each receiver of stream s, one buffer is being filled while the other one is being drained. In other words, the mobile receivers use one buffer for receiving data and the other one for decoding data and the uses of the buffers are exchanged when the next time window $p+1$ starts. It is important to note that the data received by the mobile receivers during window $p-1$ is always rendered during window p. Hence the time length of window p depends on the number of frames received in window $p-1$. For each window p, we use m_p^s to denote the last frame that can be included in window p of stream s without causing buffer overflow for the receiver. We let y_p^s denote the total amount of data to be received in window p for stream s, x_p^s to denote the start time of window p and z_p^s to denote the end time of window p. These quantities are calculated using the following equations:

$$\begin{cases} m_p^s = 0 & p = 0, \\ \sum_{j=m_{p-1}^s+1}^{m_p^s} \sum_{q=0}^{Q_s} l_{j,s,q} \leq \frac{B}{2} & \\ \quad < \sum_{j=m_{p-1}^s+1}^{m_p^s+1} \sum_{q=0}^{Q_s} l_{j,s,q} & \forall\, 1 \leq p \leq p_s. \end{cases} \quad (3)$$

$$y_p^s = \sum_{j=m_{p-1}^s+1}^{m_p^s} \sum_{q=0}^{Q_s} l_{j,s,q}. \quad (4)$$

$$x_p^s = \begin{cases} 0 & p = 1, \\ (m_{p-2}^s + 1)/F & 2 \leq p \leq p_s. \end{cases} \quad (5)$$

$$z_p^s = \begin{cases} \sum_{s=1}^{S} y_1^s / R & p = 1, \\ m_{p-1}^s / F & 2 \leq p \leq p_s. \end{cases} \quad (6)$$

We schedule the bursts for each video stream at specific points called decision points. A decision point can be a time when a new window is started (x_p^s), when a window reaches the end of its decoding time (z_p^s), or when a window has been allocated its required aggregate data amount (y_p^s). A window p is considered to be outstanding if it has not been allocated all of its required data amount y_p^s. At each decision point t, we schedule a burst for the window p that has the smallest end time z_p^s among all outstanding windows p' that have start time $x_{p'}^s \leq t$ and end time $z_{p'}^s \geq t$, and we schedule a burst for the selected window p until the next decision point. We continue to select windows and schedule bursts until there are no outstanding windows. We then wait until the next decision point to schedule new bursts. At each decision point that is the end time of a window p of a stream s, we check whether or not all of the data for this window has been scheduled. If there is remaining data for window p of video stream s, we consider all video streams s' that have bursts scheduled in the time interval from x_p^s to z_p^s. Suppose $a_p^s(s')$ and $d_p^s(s')$ are the first and the last frames of video stream s' that are scheduled in the time window p of video stream s. We want to maximize the quality of the transmitted data while respecting the bandwidth limitations. To decide which MGS quality layers to transmit, we need to calculate the transmission cost of each quality layer.

For each video stream s, we use $w_{i,i'}^s(q)$ to denote the usefulness of quality layer q from frame i to frame i', $1 \leq i \leq i' \leq I$ of video stream s. The usefulness $w_{i,i'}^s(q)$ is defined by the following equation:

$$w_{i,i'}^s(q) = \begin{cases} \infty & q = 0, \\ \frac{(\sum_{j=i}^{i'} \lambda_{j,s,q})/(i'-i+1)}{\sum_{j=i}^{i'} l_{j,s,q}} & q > 0. \end{cases} \quad (7)$$

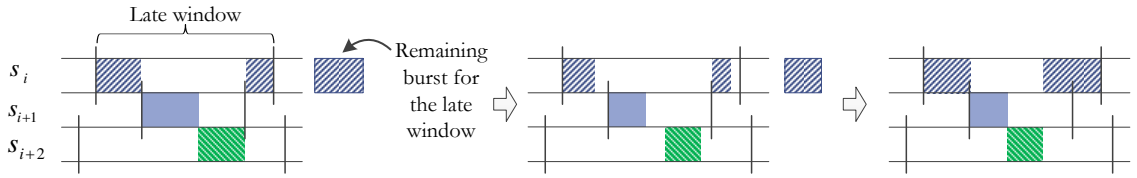

Figure 1: When a window is late, MGS layers from other bursts are dropped to accommodate the remaining data of that late window.

The usefulness $w^s_{i,i'}(q)$ can be seen as the slope of the R-D curve, which enables us to drop MGS layers in the R-D optimized way.

Fig. 2 gives our new multiplexing algorithm which we call Optimal Scalable Video Multiplexor (OSVM). The algorithm consists of two main steps. First, in lines 1–3 and 10–11, we try to transmit bursts for each video stream in time windows of dynamic lengths based on Eqs. (5) and (6). Video bursts of the same size may represent totally different playout durations as a result of the bitrate variability of video streams and this dynamic window allocation helps us to adapt to the bitrate variations in real time. But as we discussed in Sec. 1 the aggregate bitrate of video streams may exceed the bandwidth limit at some points. In this case, we might not be able to meet the data requirements for a window, which is referred to as a *late* window in line 4. In the second step, between lines 4 and 9, we consider dropping some MGS layers from the video streams in the time interval of the late window to free some bandwidth in that time interval. This gives us the opportunity to assign more data to the late window in order to make it complete, i.e., transmitting at least the base layer of the frames for basic video quality. This process is shown in Fig. 1. In this illustrative example, a window of stream s_i is late and as a result, MGS layers from other bursts for this window and all the bursts which are scheduled within the late window are inspected, and parts of the bursts are dropped in an R-D optimized fashion using Eq. (7). This frees up some broadcast bandwidth to accommodate bursts from the late window.

Complexity Analysis. In line 2 of the pseudocode, at each decision point, we check the windows of S video streams and schedule a burst for each stream. Let Q be the maximum Q_s among all video streams. If the total number of windows for all video streams is P, then the number of decision points is $P + S$. Hence, the process of scheduling bursts for the video streams takes time $O(PS + S^2)$. Constructing the windows for S video streams in lines 1 and 10 takes time $O(\sum_{s=1}^{S} I)$. Since the buffer size B for mobile devices and, as a result, the number of frames for each window are bounded by constants, constructing the windows takes $O(PS)$ time. Finally, for each late window, calculating $w^s_{i,i'}(q)$ for all $1 \le i \le i' \le I,\ 1 \le s \le S,\ 1 \le q \le Q$ takes time $O(ISQ)$, and so the total complexity for all windows is $O(PISQ)$. Since the number of MGS layers and the number of frames for each window are both bounded by constants, we can rewrite $O(PISQ)$ as $O(PS)$. Hence, the OSVM algorithm has a polynomial time complexity of $O(PS + S^2) + O(PS) + O(PISQ) = O(PS + S^2)$.

1. **generate** the first window $<x^s_1, y^s_1, z^s_1>$ for all s
2. **foreach** decision point of window p for stream s {
3. **schedule** a burst between times t and t_n for s, where p has the smallest z^s_p among all windows p' with $x^s_{p'} \le t < z^s_{p'}$ and t_n is the next decision point
4. **if** window p of s is late
5. **foreach** stream s' with bursts between x^s_p and z^s_p
6. **foreach** layer q scheduled for s'
7. **compute** $w_{s',q} = \begin{cases} w^{s'}_{a^s_p(s'),a^s_p(s')+m^s_p}(q), & s' = s \\ w^{s'}_{a^s_p(s'),d^s_p(s')}(q), & \text{o.w.} \end{cases}$
8. **sort** layers of all streams in ascending order of $w_{s,q}$
9. **drop** layers sequentially until p is complete
10. **generate** a new window for s
11. }

Figure 2: The OSVM algorithm.

6. EVALUATION

6.1 Setup

We have evaluated our algorithm using a mobile TV testbed that we developed in our lab. The base station is a Linux box with an RF signal modulator. This modulator implements the physical layer of the mobile broadcast protocol and is connected to an indoor antenna to transmit DVB-H compliant signals. We have designed and implemented a software package for the base station. We have implemented the OSVM algorithm, and we have also implemented the MBS (Nokia Mobile Broadcast Solution, see Sec. 3) and SMS [8] algorithms for comparison. We compared our new algorithm against the MBS and SMS algorithms because they are the cutting-edge multiplexing algorithms in the field and in the literature, respectively.

For the experiments, we have set the modulator to use the 16-QAM (Quadrature Amplitude Modulation) scheme and a 10 MHz radio channel, and we set the overhead to $T_0 = 100$ msec. We used ten videos, each of five minutes duration, with average bitrates ranging from 250 to 768 kbps. The video files are from the categories of sport, talk show, documentary, and TV game show and have very different visual characteristics. We encoded these ten videos into scalable streams using the Joint Scalable Video Model (JSVM) [12], which is the reference software for H.264/SVC. Each scalable stream consists of eight MGS layers. We created the trace file for each video using `BitStreamExtractorStatic` provided by JSVM. Using this tool, we extracted the size of the packet for each MGS layer of every frame. We also used this tool to extract video streams with different numbers of MGS layers from the original video. We then used `PSNRStatic` of JSVM to determine the PSNR value of each MGS layer of each frame. Based on this information, we created a trace file for each video stream, and used it in our experiments. In the trace file, there are Q_s entries for each frame i of video stream s, with entry k, $1 \le k \le Q_s$, indicating the size and the PSNR value of layer k of frame i.

We used the base station to concurrently broadcast all ten video streams for five minutes, and we stored the transmission logs, which consist of the detailed information of each burst, including timestamp, size, associated stream, and the frames and layers in the burst. We repeated the experiments with the different algorithms: OSVM, SMS, and MBS. As mentioned in Sec. 3, the MBS algo-

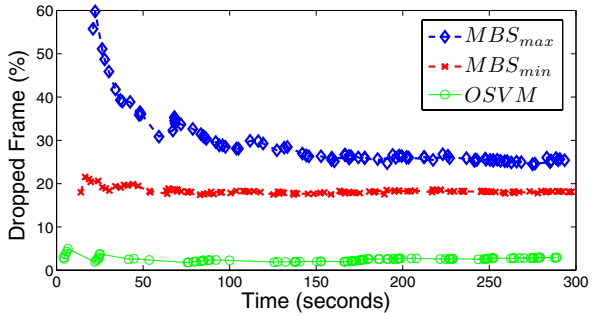

Figure 3: Dropped frames in the MBS and OSVM algorithms.

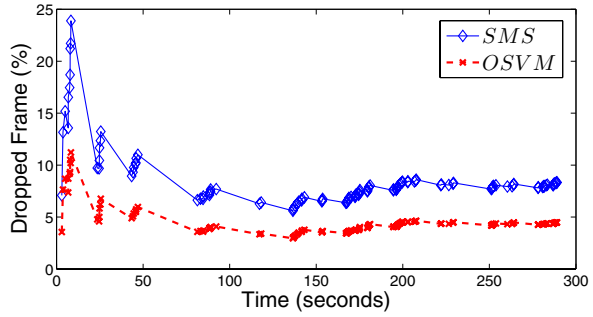

Figure 4: Dropped frames in the SMS and OSVM algorithms.

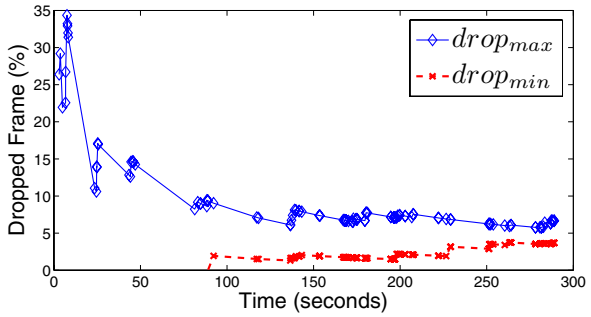

Figure 5: Maximum and minimum dropped frames among all video streams achieved by the OSVM algorithm.

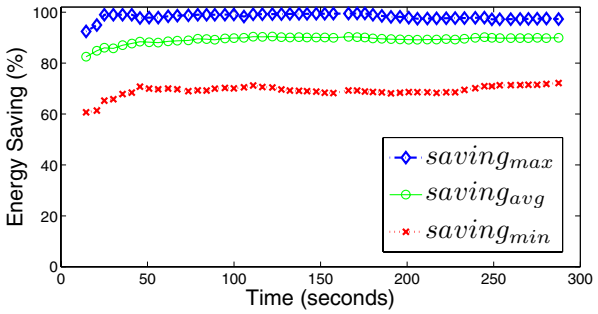

Figure 6: Maximum and minimum energy saving among all video streams achieved by the OSVM algorithm.

rithm requires manual configuration of the r_s values, which is time-consuming and error-prone. In our experiments, we varied the r_s value in the range $0.33 \times ravg_s$ to $3 \times ravg_s$, where $ravg_s$ is the average bitrate for video stream s. We conducted our experiments with all considered r_s values, and we report the best performance among them. That is, we exhaustively searched for the *absolutely optimal* MBS configuration, and we compare the OSVM algorithm to it. We consider three performance metrics: (i) frame-drop ratio, (ii) energy saving, and (iii) video quality in PSNR.

6.2 Results

Frame-Drop Rate. We compare the OSVM algorithm against the MBS method used in current mobile broadcast networks. We plot the average frame-drop rate in Fig. 3. In this figure, MBS_{min} and MBS_{max} are the best and worst results from the MBS method. In particular, each point on the MBS_{min} curve is the minimum over all ten streams at that time and the point on the MBS_{max} curve is the maximum. Thus the two curves define the envelope containing all points of all ten streams. The figure shows that OSVM significantly outperforms even the best case of the MBS method.

Next, we compare OSVM to the SMS algorithm in Fig. 4. Due to the VBR nature of the video streams, the drop rate varied from 6% to 21% for SMS and from 3% to 9% for OSVM at different points in time, but the averages, over time, were 8.35% for SMS and 4.49% for the OSVM algorithm. Stated another way, the OSVM algorithm drops an average of 46.2% fewer frames than the SMS algorithm, which leads to smoother playout. This improvement is achieved by using our weighting model for MGS layers and our decision mechanism described in Sec. 5 to select the layers for each video stream that maximize the number of transmitted frames while maintaining high per-frame quality over all video streams.

Fig. 5 shows the envelope for dropped frames for all video streams for the OSVM algorithm. At the beginning, the percentage of dropped frames varies between 0% and 35% but it quickly converges to a range of 4.49% to 6.6%. This figure shows that the low average frame-drop rate is the result of good performance among all video streams.

Energy Saving. In Fig. 6, $saving_{avg}$ shows the average energy saving resulting from use of the OSVM algorithm based on Eq. (2b), and $saving_{min}$ and $saving_{max}$ define the envelope of the energy saving achieved among all video streams. The energy saving for the video streams ranges from 70% to 99% and the average is 90%. This good energy saving is achieved despite the fact that our primary objective, maximizing bandwidth utilization, can conflict with the energy saving objective.

Video Quality. We compute the PSNR values for the SMS and OSVM algorithms. In our analysis, we consider each dropped frame as a blank image to compute its PSNR value. Fig. 7 shows that the average PSNR value resulting from OSVM is an improvement of about 1.34 dB over the SMS algorithm. In addition, we found that the average PSNR value resulting from the OSVM algorithm is only about 1.17 dB lower than the average PSNR of the original video stream ($PSNR_{max}$ in Fig. 7). That is, relative to $PSNR_{max}$, OSVM causes only half as much PSNR drop as the SMS algorithm.

We also plot the maximum and minimum video quality among all video streams in Fig. 8, which are labelled $PSNR_{min}$ and $PSNR_{max}$, respectively. This figure shows that the PSNR values of individual video streams converge to a small range of 41.92 to 42.98 which is very close to the average of 42.60. This small

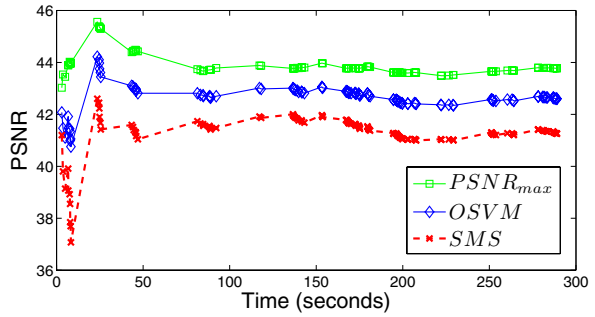

Figure 7: Video quality achieved by the SMS and OSVM algorithms.

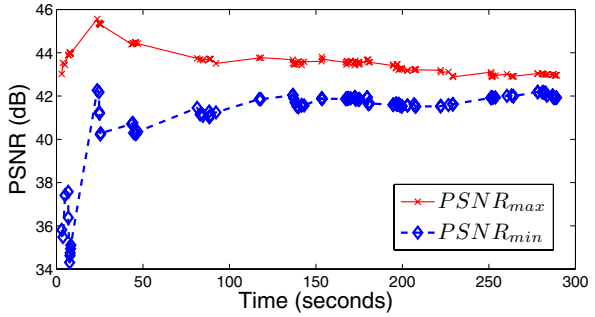

Figure 8: Maximum and minimum PSNR among all video streams achieved by the OSVM algorithm.

gap, merely 1.0 dB, shows that the OSVM algorithm leads to high video quality for all video streams, rather than a subset of them.

We would like to point out that the OSVM algorithm was running in real-time on a base station loaded with the control software and video streaming servers for broadcasting. This shows the applicability of OSVM as a real-time multiplexing algorithm.

7. CONCLUSIONS

In this paper, we studied the problem of broadcasting multiple scalable video streams from a base station to a large number of mobile receivers. Our primary objective was to maximize bandwidth utilization, and our secondary objectives were to achieve close to optimal energy saving and perceived quality of the video streams. To achieve this, we proposed an algorithm that uses a model of the quality of video streams and a weighting mechanism for video frames to drop video packets whenever the aggregate bitrate of the video streams exceeds the available bandwidth. We compared our algorithm with SMS [8] and MBS [11] algorithms and our experimental results on a real mobile TV testbed show that the OSVM algorithm significantly outperforms current methods and also achieves near optimal energy saving and video quality. For example, the OSVM algorithm leads to: (i) 46.2% fewer frames dropped compared to the SMS algorithm, (ii) 90% energy saving on average, and (iii) approximately 1.34 dB PSNR improvement compared to the SMS algorithm.

8. REFERENCES

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

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

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

[4] CAMARDA, P., TOMMASO, G., AND STRICCOLI, D. 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).

[5] Digital Video Broadcasting (DVB); DVB specification for data broadcasting. European Telecommunications Standards Institute (ETSI) Standard EN 301 192 Ver. 1.4.1, June 2004.

[6] FARIA, G., HENRIKSSON, J., STARE, E., AND TALMOLA, P. DVB-H: Digital broadcast services to handheld devices. *Proc. of the IEEE 94*, 1 (January 2006), 194–209.

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

[8] HSU, C., AND HEFEEDA, M. On statistical multiplexing of variable-bit-rate video streams in mobile systems. In *Proc. of ACM Multimedia'09* (Beijing, China, October 2009), pp. 411–420.

[9] HSU, C., AND HEFEEDA, M. Time slicing in mobile TV broadcast networks with arbitrary channel bit rates. In *Proc. of IEEE INFOCOM'09* (Rio de Janeiro, Brazil, April 2009), pp. 2231–2239.

[10] LIEBL, G., JENKAC, H., STOCKHAMMER, T., AND BUCHNER, C. Radio link buffer management and scheduling for wireless video streaming. *Telecommunication Systems 30*, 1-3 (November 2005), 255–277.

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

[12] SVC reference software. `http://ip.hhi.de/imagecom_G1/savce/downloads/SVC-Reference-Software.htm`.

[13] RAPPAPORT, T. *Wireless Communications: Principles & Practice*, 4th ed. Prentice Hall, 1996.

[14] REICHEL, J., WIEN, M., AND SCHWARZ, H., Eds. *Scalable Video Model 3.0*. ISO/IEC JTC 1/SC29/WG11, doc. N6716, Palma de Mallorca, Spain, October 2004.

[15] REZAEI, M., BOUAZIZI, I., AND GABBOUJ, M. Joint video coding and statistical multiplexing for broadcasting over DVB-H channels. *IEEE Transactions on Multimedia 10*, 7 (December 2008), 1455–1464.

[16] SCHIERL, T., STOCKHAMMER, T., AND WIEGAND, T. Mobile video transmission using scalable video coding. *IEEE Transactions on Circuits and Systems for Video Technology 17*, 9 (September 2007), 1204–1217.

[17] SCHWARZ, H., MARPE, D., AND WIEGAND, T. Overview of the scalable video coding extension of the H.264/AVC standard. *IEEE Transactions on Circuits and Systems for Video Technology 17*, 9 (September 2007), 1103–1120.

[18] SEN, S., REXFORD, J., DEY, J., KUROSE, J., AND TOWSLEY, D. Online smoothing of variable-bit-rate streaming video. *IEEE Transactions on Multimedia 2*, 1 (March 2000), 37–48.