Effective Bandwidth Based Scheduling for Streaming Multimedia

Sang H. Kang
Communication Networks Research Lab.
Department of ECE, University of Seoul
Email: shkang@uos.ac.kr

Avideh Zakhor
Video and Image Processing Lab.
Department of EECS, U.C. Berkeley
Email: avz@eecs.berkeley.edu

***Abstract*— We propose a class of packet scheduling algorithms for streaming media. The importance level of a video packet is determined by its relative position within its group of pictures, taking into account the motion-texture discrimination and temporal scalability. We generate a number of nested substreams, with more important streams embedding less important ones in a progressive manner. We model the streaming system as a queueing system, compute the run-time decoding failure probability of a frame in each substream based on *effective bandwidth*, and determine the optimum substream to be sent at any moment in time. The data within optimum substream is sent based on earliest-deadline-first scheduling, until the next channel report arrives, at which time the optimum substream is recomputed. From experiments with real video data, we show that our proposed scheduling scheme outperforms the conventional sequential sending scheme.**

I. Introduction

Streaming media applications are becoming increasingly more popular on the Internet. Properties of streaming media include pre-encoding and pre-roll buffering, which allow rate-distortion optimization and data packet scheduling to adaptively select data to be transmitted, as the channel status varies [1], [2].

Encoded media data consist of packets with different levels of importance and different impact on video quality. Therefore, it is desirable to develop a packet scheduling algorithm in a rate-distortion optimized manner. Related rigorous work on packet scheduling using rate-distortion optimization can be found in [3]. In [3], a Markov decision process is used to find optimal scheduling policy through iterations over all the packets in the scheduling window. However, the computational complexity of [3] may not guarantee performance for real-time implementation. In [4], a simplified heuristic formulation and cost function for rate-distortion optimization is proposed in order to reduce the computational complexity.

In this paper, we address the problem of packet scheduling of streaming media over a lossy network, in a rate-distortion optimized way, assuming reverse channel availability, and no contraction or expansion of the original display duration of clips. Our approach is to generate a number of nested substreams, with more important streams embedding less important ones in a progressive manner. We then determine the optimum importance level in order to divide the entire video data into low and high priority portions using the effective bandwidth approach. Our proposed scheduling algorithm consists of (a) sending the high priority portion, followed by low priority portion; (b) sending the packets within each priority, according to earliest-deadline-first (EDF) scheduling; (c) packets within the same priority and same deadline are sent according to importance levels; (d) packets deemed to have missed their deadline are not sent.

This work was completed when Sang Kang was a visiting researcher at the Video and Image Processing Lab at U.C. Berkeley. This work was supported by NSF grant ANI-9905799 and AFOSR contract F49620-00-1-0327.

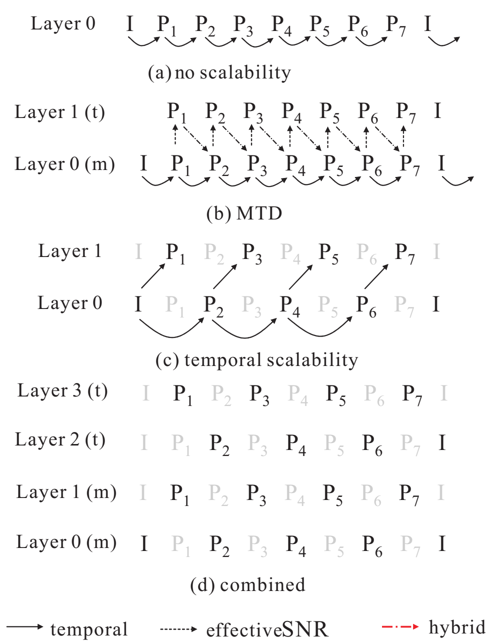

Fig. 1. Scalabilities and layer structures

II. Scalabilities and Importance Levels

Assume a video frame sequence F_n $(n = 0, 1, 2, \cdots)$ with frame rate f in frames/sec. Suppose that the receiver begins to display F_0 at time $t = 0$ sec.; then the display time or *deadline* of F_n is $t = n/f$ sec. The video sequence is divided into GOPs, where a GOP is assumed to have N frames with one I frame, and $(N-1)$ P frames. Within each GOP, the j-th P frame is labeled as P_j $(j = 1, 2, ..., N-1)$.

Assigning importance levels is closely related to the concept of scalability or layering. We assume that data in a frame is given by only one importance level except for MTD case, in which data in a P frame is divided into two importance levels according to motion and texture. In Fig. 1, we show directed dependency graphs for a number of layering structures for scalable encoding. In each layering scheme, higher importance is assigned to lower layers first, and within a layer, data corresponding to smaller frame numbers, appearing earlier in the frame, are of higher importance than larger frame numbers.

Fig. 1(a) shows a dependency graph typical of encoding of a group of frames packetized sequentially, without any scalability, in which we assign importance $h = 0$ to I frames, $h = 1$ to P_1 frames, and so on. In this case we have a total of $H = N$ importance levels with N frames in a GOP.

In Fig. 1(b), we show the layering structure for MTD, where I frames and motion part of P frames form the base layer, and texture part of P frames, the enhancement layer. Characters 'm' and 't' in the parentheses denote motion and texture, respectively. We assign higher importance level to the motion part than texture. In this case, there are a total of $H = 2N - 1$ importance levels.

We also apply our scheduling algorithm to a two-layer scheme shown in Fig. 1(c). In this case the total number of importance levels is $H = N$, which is the same as in the non-scalable case in Fig. 1(a). I frames and even numbered P frames form the base layer, and odd numbered P frames form the enhancement layer.

Combination of spatial and temporal scalability has been investigated in [7]. We consider the combination of MTD and two-layer temporal scalability as shown in Fig. 1(d). For simplicity, we have deliberately not drawn the dependency arrows between different frames. In this case, the total number of importance levels is $H = 2N - 1$ with importance decreasing from layer 0 to 1, to 2, to 3.

III. Packet Scheduling based on Effective Bandwidth

The data in each importance level in a GOP is segmented into fixed-size *packets* for transmission. Define a random process $\{X_i^{(h)}; i = 0, 1, \cdots\}$ to be the sequence of the size, in packets, of data with importance level h in the i-th GOP, assumed to be wide sense stationary. Also define $S_i^{(\theta)}$ $(\theta = 0, 1, \cdots, H-1)$ to be the partial sum such that

$$S_i^{(\theta)} = \sum_{h=0}^{\theta} X_i^{(h)}. \tag{1}$$

Autocovariance of $S_i^{(\theta)}$ is defined as

$$v^{(\theta)}[k] = \mathrm{E}[(S_i^{(\theta)} - \mathrm{E}[S^{(\theta)}])(S_{i+k}^{(\theta)} - \mathrm{E}[S^{(\theta)}])], \qquad k = 0, 1, \cdots. \tag{2}$$

Let $\Gamma^{(\theta)}$ denote a nested substream, in which each GOP is composed of data with importance levels $h = 0$ through $h = \theta$, for the entire video sequence. Assuming GOP duration of 1 sec., the data size of i-th GOP in $\Gamma^{(\theta)}$ is represented by $S_i^{(\theta)}$, and the average data rate of $\Gamma^{(\theta)}$ is $\mathrm{E}[S^{(\theta)}]$ in packets/sec.

For substream $\Gamma^{(\theta)}$, we introduce a queueing model, as shown in Fig. 2, with $\Gamma^{(\theta)}$ as the stochastic input process, an output service rate equal to the channel bandwidth C packets/sec., and a fixed-size buffer of size $B^{(\theta)}$. The buffer size $B^{(\theta)}$ of the queueing model needs to be determined carefully, taking into account the receiver buffer status. Specifically, the buffer size, $B^{(\theta)}$, is determined from the pre-roll video duration for substream $\Gamma^{(\theta)}$, in the receiver buffer, which is represented in terms of the number of packets according to the average data rate of $\Gamma^{(\theta)}$. In our approach, we relate $B^{(\theta)}$ to the number of GOPs for the data with importance levels of $h = 0$ through $h = \theta$ as follows. Suppose that the receiver is currently decoding a frame in GOP number k, and the largest GOP number, for which all packets with importance θ have so far been acknowledged, is measured to be m_θ $(m_\theta \geq k)$. If the GOP duration is 1 second, this means that the receiver has a $(m_\theta - k)$ seconds long queue in its buffer for substream $\Gamma^{(\theta)}$. To represent the buffer size corresponding to $(m_\theta - k)$ seconds in terms of the number of packets in $\Gamma^{(\theta)}$ whose average rate is $\mathrm{E}[S^{(\theta)}]$ packets/sec., we propose to determine $B^{(\theta)}$ as:

$$B^{(\theta)} = (m_\theta - k)\mathrm{E}[S^{(\theta)}] \qquad \text{[packets]}. \tag{3}$$

Given substream $\Gamma^{(\theta)}$ whose average data rate $\mathrm{E}[S^{(\theta)}]$ is less than the channel bandwidth C in the queueing model in Fig. 2, the input

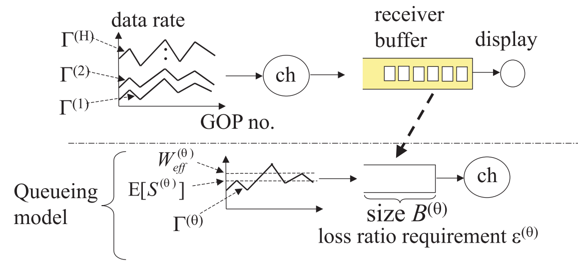

Fig. 2. Queueing system model for streaming media

process $\Gamma^{(\theta)}$ suffers from packet losses due to video rate fluctuations. Applying the effective bandwidth notion [5] to our problem, given channel throughput C, an estimation of the packet loss probability for $\Gamma^{(\theta)}$ in the queueing model is given by:

$$\epsilon_p^{(\theta)} = e^{\beta}, \tag{4}$$

where

$$\beta = \frac{-2B^{(\theta)}(C - \mathrm{E}[S^{(\theta)}])}{\sum_{k=0}^{\infty} v^{(\theta)}[k]}. \tag{5}$$

$\epsilon_p^{(\theta)}$ increases when $B^{(\theta)}$ becomes small and/or the video fluctuations become large. Also, $\epsilon_p^{(\theta)}$ increases when $\mathrm{E}[S^{(\theta)}]$, the average data rate of $\Gamma^{(\theta)}$, becomes large. However, to ensure $\epsilon_p^{(\theta)}$ remains in the range [0,1], we must have $\mathrm{E}[S^{(\theta)}] < C$.

Within $\Gamma^{(\theta)}$, we define the decoding failure of a GOP as loss of any motion packet in the GOP. Let $\epsilon_{GOP}^{(\theta)}$ denote GOP decoding failure probability for $\Gamma^{(\theta)}$ in the queueing model. The probability $\epsilon_{GOP}^{(\theta)}$ is approximated as follows [8]:

$$\epsilon_{GOP}^{(\theta)} = \begin{cases} 1 - (1 - \epsilon_p^{(\theta)})^{\mathrm{E}[S^{(\theta)}]}, & \theta \leq N - 1 \\ 1 - (1 - \epsilon_p^{(\theta)})^{\mathrm{E}[S^{(N-1)}]}, & \theta > N - 1 \end{cases} \tag{6}$$

From queueing theory, we note that $\epsilon_{GOP}^{(\theta)}$ is a monotonically increasing function of θ. The above GOP decoding failure probability is compared with a given user specified requirement in order to determine optimum θ in our scheduling algorithm.

Suppose it is required that the GOP decoding failure probability to be less than γ. Given C, the server determines the optimum importance level θ_o such that:

$$\theta_o = \arg\max_{\theta} \left\{ \epsilon_{GOP}^{(\theta)} < \gamma \;\text{ and }\; \mathrm{E}[S^{(\theta)}] < C \right\}. \tag{7}$$

Thus the optimum importance level is the largest θ such that the frame decoding failure probability of substream $\Gamma^{(\theta)}$ is less than the requirement γ, while keeping the expected bit rate of the stream below channel throughput, C. As stated earlier, this second condition is needed to ensure $\epsilon_p^{(\theta)}$ is in the range [0,1]. Since θ_o is constrained to be an integer, in an ideal scenario, the server must alternate between θ_o and $\theta_o + 1$ in order to regulate the receiver buffer, and to maximize utilization of the channel bandwidth. Specifically, the relative time spent at θ_o versus $\theta_o + 1$ has to do with receiver buffer fullness; when receiver buffer is full $\theta_o + 1$ is chosen, whereas when it is empty θ_o is chosen.

Given θ_o, the importance levels are divided into two priorities; high priority part with importance levels from $h = 0$ through $h = \theta_o$, and low priority, from $h = \theta_o + 1$ through $h = H - 1$. High priority packets are scheduled to be transmitted first, followed by low priority packets. Within each priority, packets are sent according to EDF.

To make the scheduling algorithm be adaptive to time-varying channel fluctuations and queue status in the receiver buffer, the receiver reports measured throughput periodically to the server, which then computes mean throughput as an exponentially weighted moving average (EWMA) as follows:

$$C \leftarrow \alpha C + (1-\alpha) Y_m, \tag{8}$$

where $0 < \alpha < 1$, and Y_m is the newly reported throughput. For packet switched networks, the sending rate is based on TCP-friendly rate based equation [6], thus requiring estimation of round trip time and packet loss probability.

In order to take into account the variability of the channel throughput, we also assume that the sender calculates the variance σ_Y^2 of Y_m. Intuitively, given average channel throughput, the video quality at the receiver will be more degraded when the channel fluctuation, i.e., σ_Y^2, becomes large. Since channel fluctuations and video rate fluctuations are statistically independent, we can lump them together to obtain:

$$\beta = \frac{-2B^{(\theta)}(C - \mathrm{E}[S^{(\theta)}])}{\sum_{k=0}^{\infty} v^{(\theta)}[k] + \sigma_Y^2} \tag{9}$$

in calculating optimum θ at the server.

IV. Performance Evaluation

Our proposed scheduling method is evaluated by sending real video data on wire-line and wireless channels. We use talk show *Larry King Live* and movie trailer *Preacher's Wife*. Both videos are in QCIF format and encoded as 15 frames/sec. Talk show video has 1689 frames, and movie trailer, 1871 frames. We use Microsoft MPEG-4 Visual Reference Software version 2 FDMA1-2.3-001213, to encode/decode video data. We have added error recovery/concealment features related to MTD and temporal scalability. Our proposed packet scheduling algorithm is implemented on top of the transport layer using UDP. For each arriving 1000-byte packet, the receiver sends a 3-byte acknowledgment including time stamp and measured throughput. Based on the time stamp, the sender estimates RTT and obtains the maximum RTT over time, denoted by $\mathrm{RTT}_{\max}$. Once a packet is sent, the packet is referred to as outstanding and a timer is set to go off after $\mathrm{RTT}_{\max}$ seconds. If the acknowledgment arrives before the timer goes off, then it is referred to as an acknowledged packet. If the timer goes off without acknowledgment, then the packet becomes a not-sent packet to be retransmitted. Note that the sender does not distinguish between retransmission packets and first-time transmission ones.

Our proposed effective bandwidth scheduling (EBS) is compared with a conventional sequential sending (SS). In both schemes, the sender drops the remainder of a GOP as soon as it finds a frame that has missed its deadline. The receiver begins display after its buffer is filled with 10 seconds worth of video. For our optimization scheme, we use, $\gamma = 10^{-4}$, as the decoding failure probability requirement for high priority part, and EWMA factor of $\alpha = 0.9$.

First, we transmit talk show video over the wire-line Internet between Korea Advanced Institute of Science and Technology (KAIST) and the University of California, Berkeley. The average round trip time was measured to be 250 msec - 300 msec. In our experiments, we repeatedly transmit the video clip to increase time duration of our experiment. We use talk show video with average video bit rate of 125.5 Kbps. Since the real channel bandwidth of the wired Internet is measured to be far larger than the video bit rate, we have implemented ON/OFF type artificial error in the receiver to achieve average channel throughput of 60% to 95% relative to the average video bit rate, 125.5 Kbps. In Fig. 3, we illustrate the distortion curves for various schemes as a function of channel throughput. In the legend, TS stands for temporal scalability. EBS outperforms SS

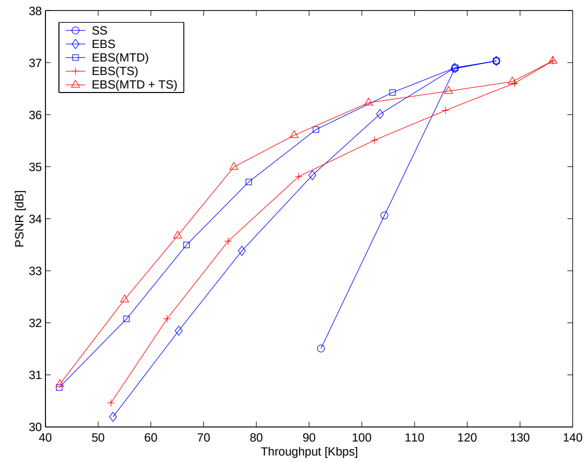

Fig. 3. Rate-distortion curves with various schemes

at all channel throughputs. The curves with triangles and crosses are obtained using two-layer temporal scalability (TS), which has been observed to increase the bit rate by 10%. Therefore, with channel throughput of 125.5 Kbps, they show degradation of about 0.5 dB as compared with all other schemes. At channel throughputs below 95 Kbps, MTD combined with TS scheduling outperforms all other schemes, and MTD alone scheduling outperforms TS alone scheduling. To summarize, except at 125.5 Kbps, at all other channel throughputs, there exists at least one EBS that outperforms SS.

In Fig. 4, we compare SS and EBS, showing sample paths of various statistics as a function of time. The results correspond to channel throughput of 67 Kbps or 53% of the video rate. For EBS, we use MTD with $N = 16$ frames in a GOP resulting in total of $H = 31$ importance levels. After initial buffering for 10 seconds, the receiver starts display at time $t = 0$. Fig. 4(a) shows video bit rate fluctuations. The channel fluctuation is shown in Fig. 4(b). In SS, the server tries to deliver data with all levels of importance with same priority, i.e., the threshold of importance level is constant at $\theta_o = H - 1 = 30$, over time as shown in Fig. 4(c). With EBS however, θ_o is determined adaptively according to the channel throughput fluctuations and queue length of the receiver buffer as seen in Fig. 4(c). In Fig. 4(d), we compare performance of the two schemes in terms of PSNR, reported every 5 seconds. SS shows sharp drops when video bit rate is consistently high and/or the channel error is heavy. In some regions, SS shows better PSNR than the scheduling scheme which is trying to save bandwidth by sending future important data in advance. Overall, the proposed scheme outperforms SS by smoothing out the bursty distortion periods and resulting in less variability in PSNR–an important factor from a human perception point of view. Furthermore, average PSNR for EBS is 4.2 dB higher than SS.

In Figs. 4(e) and 4(f), we illustrate the receiver buffer status as a function of time, reported every second. The vertical axes in Figs. 4(e) and 4(f), denote the number of GOPs in the receiver buffer with at least the I frame. The number of importance levels in each of those GOPs is illustrated in grayscale for SS and EBS in Figs. 4(e) and 4(f), respectively. Specifically, a GOP that only has data with $h = 0$ in the receiver buffer is almost white, a GOP with all levels of data in the receiver buffer is black, and GOPs with the number of importance levels in between these two extremes are shown in gray according to the legend in Fig. 4(g). Note that the number of GOPs increases when new packets arrive at the receiver,

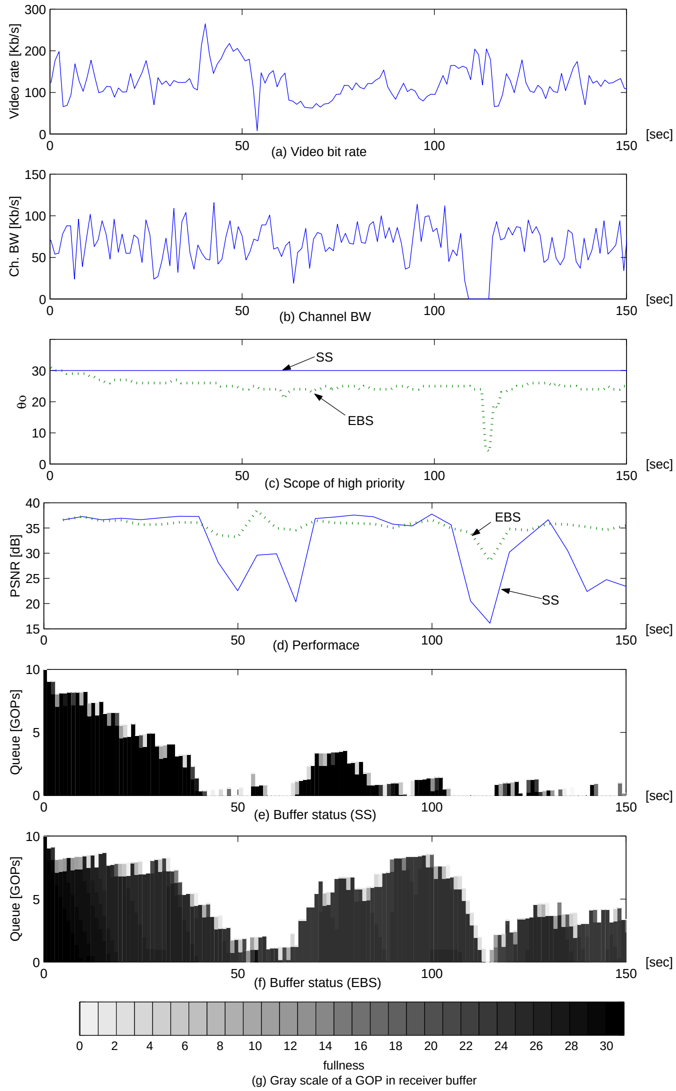

Fig. 4. Sample paths of various statistics over time in streaming talk show video

and decreases by one every $N/f = 16/15$ seconds, where N/f is a GOP duration in our experiments. Initially receivers have 150 frames, i.e., approximately 9 GOPs, corresponding to 10 seconds worth of data in their buffer. Since SS sends all levels of importance, it fills the receiver buffer with *black* GOPs with large number of importance levels, resulting in immediate exhaustion of the initial pre-roll buffer. On the other hand, EBS regulates the queue length in the receiver buffer with optimum scope of importance levels, shown as grayscale GOPs in Fig. 4(f). Our scheme effectively absorbs the video fluctuation at time around 50 second, and severe channel fluctuations at time around 110 second.

We now consider wireless channels such as wireless LAN. In Fig. 5, we show PSNR performance for sending talk show and movie trailer clips over IEEE 802.11 wireless LAN. Generating background cross traffic of 2.1 Mbps, the server opens two sessions with the client at the same time, one session with SS and the other with EBS using MTD; this way, channel congestion and/or interference have the same effect on both schemes. As seen in Fig. 5, the scheduling scheme shows significantly larger improvement over SS than in the previous

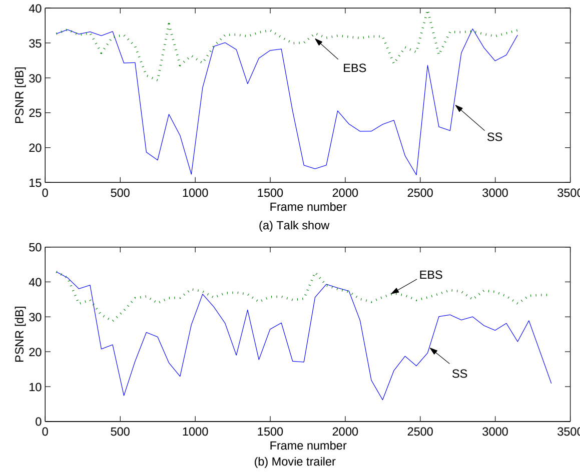

Fig. 5. Performance on wireless LAN

wire-line network case. The average PSNR for SS and EBS are 27.0 dB and 35.8 dB, respectively, for talk show in Fig. 5(a), and 22.1 dB and 36.2 dB, respectively, for movie trailer in Fig. 5(b). In each curve, video quality in the beginning is good due to initial pre-roll buffering. However, as time goes on, SS scheme shows degradation since the receiver buffer is empty most of the time, whereas EBS shows consistentely high video quality.

V. Conclusion

We have developed a class of rate distortion optimized packet scheduling algorithms, taking into account both video bit rate fluctuations as well as channel fluctuations. In doing so, we take advantage of temporal scalability and MTD in order to arrive at a set of nested substreams whereby the more important substreams embed the less important ones. We apply the notion of effective bandwidth from queueing theory to determine the optimum substream to send at any moment in time. We have shown the effectiveness of our approach over sequential sending scheme for actual video sequences over both the wire-line Internet and WLAN. Future work involves optimization over combined networks from wire-line and wireless links, optimization with VCR functionality, and applying EBS to multicast networks.

References

[1] H. Radha, Y. Chen, K. Parthasarathy, and R. Cohen, ”Scalable Internet video using MPEG-4,” *Signal Processing: Image Communication,* vol. 15, no. 1–2, pp. 95 – 126. Sept. 1999.

[2] G. J. Conklin, *et. al.*, ”Video coding for streaming media delivery on the Internet,” *IEEE Trans. Circuits and Systems for Video Technology,* vol. 11, no. 3, pp. 269 – 281, Mar. 2001.

[3] P. A. Chou and Z. Miao, ”Rate-distortion optimized streaming of packetized media,” submitted to *IEEE Trans. Multimedia,* Feb. 2001.

[4] Z. Miao and A. Ortega, ”Expected run-time distortion based scheduling for delivery of scalable media,” *Int’l Packetvideo Workshop 2002,* Apr. 2002.

[5] C. Courcoubetis and R. Weber, ”Effective bandwidths for stationary sources,” *Prob. Eng. Inf. Sci.* vol. 9, pp. 285 – 296, 1995.

[6] S. Floyd, M. Handley, J. Padhye, and J. Widmer, ”Equation-based congestion control for unicast applications,” *ACM SIGCOMM 2000,* pp. 43 – 56, Aug. 2000.

[7] F. Wu, S. Li, R. Yan, X. Sun, and Y.-Q. Zhang, ”Efficient and universal scalable video coding,” *IEEE ICIP 2002*, vol. 2, pp. 37 – 40, 2000.

[8] Sang H. Kang and A. Zakhor, ”Rate-distortion optimization and packet scheduling based on effective bandwidth for streaming of scalable media,” *IEEE Trans. Multimedia,* 2002. Submitted.