An Integrated Source Coding and Congestion Control Framework for Video Streaming in the Internet

Kang-Won Lee* Rohit Puri† Tae-eun Kim* Kannan Ramchandran† Vaduvur Bharghavan*
* Coordinated Science Laboratory, University of Illinois at Urbana-Champaign
† Electrical Engineering and Computer Science, University of California at Berkeley
{kwlee, tkim, bharghav}@timely.crhc.uiuc.edu {rpuri, kannan}@eecs.berkeley.edu

Abstract—We describe a framework for video transmission over the Internet that features the coordinated operation of an application-layer video source coding algorithm and a transport-layer rate control mechanism. The proposed video coding scheme operates on a successively encoded video stream and provides graceful resilience to network packet drops. The robustness is enabled through a generalized Multiple Description (MD) coding strategy, architected as an adaptive array of packet-erasure correction codes. The video coding algorithm is matched to an efficient and reactive rate control mechanism that minimizes the fluctuation of rate and uses the profile of past losses to adjust the rate in a TCP-friendly manner.

While the two constituent algorithms identified above are interesting in their own right, a key feature of this work is the integration of these algorithms in a simple framework that seeks to maximize the expected delivered video quality at the receiver through coordinated adaptation of the two components. We present simple simulation results to illustrate the utility of our approach.

I. INTRODUCTION

In the current deployment of the Internet, routers in the network are oblivious to the structure or content of the packets that traverse through them. Typically, tail-drop routers discard incoming packets when output queues are full, and random-drop routers discard packets at random as the output queues start to fill up. The sender estimates the available connection capacity by probing the network, i.e., periodically incrementing the transmission rate till some router drops a packet due to queue overflow. Upon detecting this loss via end-to-end feedback from the receiver, the sender throttles the transmission rate by a large multiplicative factor (typically 0.5) [1], [2], [3].

This network design is inappropriate for video packet flows because of two reasons. First, typical video-encoded bitstreams are highly structured, e.g., characterized by a hierarchy of importance layers. The design of multiresolution (MR)-based scalable/layersed/progressive encoders is becoming increasingly popular in the source coding community for both still images and motion video [4], [5], [6]. However, routers may drop newly arriving higher priority packets in favor of already-queued lower priority packets, and may deliver more partial frames rather than fewer complete frames. Second, most current congestion control algorithms follow the “probe-lose-throttle” principle and hence experience significant variations in instantaneous rate even if the network resources available for the connection remain invariant. This can lead to a potentially high variance in the delivered video quality unless the sender accurately estimates the average sending rate, which is difficult to accomplish in dynamic network conditions. In summary, the current design of the Internet is best suited for transporting non-prioritized packet flows that are insensitive to rate fluctuations and to the relative position of packet losses. There is thus an inherent mismatch between the end-to-end requirements of the video source coding algorithms and the capabilities of the network as it is deployed today.

This underlines the need for an efficient transcoding mechanism that converts the MR-based prioritized bitstream into a non-prioritized packet stream while ensuring graceful quality degradation when there is a packet loss. In this paper, we propose a transcoding mechanism based on Multiple Description (MD) robust source coding principles, anchored on Forward Error Correction (FEC) channel codes.

From the end user’s viewpoint, a medium but constant quality is more desirable than a high and jittery one. For a smooth video reception, we want the variation of the sender’s transmission rate to be as small as possible. However, the conventional linear increase multiplicative decreased (LIMD)-based congestion control causes large fluctuations in the transmission rate even in the absence of congestion. In this paper, we propose an augmentation to LIMD congestion control that achieves low rate fluctuations (hence is more suited to streaming applications) when the connection capacity is invariant without compromising the responsiveness to the onset of congestion.

While the source transcoding algorithm and the rate control algorithm are useful contributions in their own right, a key aspect of our approach is the coordination of the source coding and rate control algorithm in an integrated architecture. In the proposed system, the source transcoding module and the congestion control module exchange connection state in a very simple yet efficient manner.

To summarize, the following are the three contributions of the paper.

1. In the source transcoding layer, we propose a simple and fast method of transforming an MR bitstream into an MD packet stream using efficient erasure codes [7], which is optimal in the sense that the delivered quality at the receiver is maximized when the channel condition is described accurately.

2. In the transport layer, we propose a TCP-friendly rate control algorithm that reduces rate fluctuation when the connection capacity is invariant and at the same time quickly reacts to changes in the connection capacity.

3. The ‘glue’ that holds the source layer and the transport layer together is the interaction layer, which packages the connection state in a way that can be used effectively by the source layer. In the proposed system, the transcoder requires the transmission profile of the connection, and the rate budget. This information is inferred by the interaction layer by monitoring the progress of the connection in the kernel space, and efficiently (without context switches) passed up to the user space.

---

1 Examples of important video signal descriptors include anchor-frame data (e.g. MPEG I-frames), coarse information such as DC and low-frequency AC transform data, and that of less important information include high-frequency AC transform data and perceptually-insignificant textured areas in the picture, etc.

2 MD coding refers to the generation of multiple independent and unprioritized representations of the same source stream having the property that the reconstruction quality is proportional to the number of representations available at the decoder.
The goal of our work is to exploit the power of shared information between the transport layer and the source coding layer – this approach has also been explored in several recent works, e.g. using different constituent components in [8] and using a layered coding framework in [9]. Figure 1 describes a simple block diagram of the system, and the components proposed in this paper.

The remainder of this paper is organized as follows: Section II describes the transcoding scheme. Section III presents the congestion control algorithm. Section IV integrates the two components via the interaction layer. Section V evaluates the performance of our system via simulation. Section VI places our work in the context of related work, and Section VII concludes the paper and offers directions for future work.

II. FORWARD ERROR CORRECTION (FEC)-BASED MULTIPLE DESCRIPTION CODING

In this section, we first describe the transcoding mechanism that converts the prioritized multiresolution (MR) bitstream into an unprioritized multiple description (MD) stream using efficient erasure channel codes. The quality profile, i.e. the quality experienced at the receiver as a function of the number of packet losses of the generated stream has the property of graceful quality degradation with respect to packet losses at the receiver. We then describe an algorithm that optimizes the quality profile in the following sense: given a multiresolution bitstream, a rate budget (the maximum number of packets of a length L can be injected into the network) and a transmission profile (the probability of successfully delivering i packets out of N, ∀ i), the algorithm outputs a quality profile that attains the maximum expected average quality. Since we are dealing with encoding a video source having a fidelity criterion, we invoke a rate-distortion (R-D) framework.

A. Transcoding Mechanism

The quality profile reflects the target quality (or equivalently distortion d) when any k out of N descriptions are received. We use the notation \((k, d(k))\) to denote quality profile, where \(d(k)\) is the distortion vector with dimension \(m \leq N\) whose entries reflect the distortion profile. We will refer to the dimension \(m\) as order of the quality profile, i.e. the number of quality levels in the profile.

\[
\begin{bmatrix}
R_0 & R_1 & R_2 & \cdots & \cdots & R_m \\
1 & 2 & \cdots & i & \cdots & m \\
\end{bmatrix}
\]

Fig. 2. Progressive bitstream from the source coder partitioned into m layers

Given \(N\) and \(d(k)\), and a progressive bitstream, the bitstream is marked at \(m\) different positions (see Figure 2) which correspond to the attainment of the distortion levels \(d(k)\), and is thus partitioned into \(m\) sections or resolution layers. The \(i\)th layer is decodable when the number of erasures does not exceed \(m - i\).

We can achieve this by using the family of Reed-Solomon erasure-correction block codes [16], which are characterized by the optimal code parameters \((N, k, N-k+1)\), which can correct any \((N-k)\) erasures out of \(N\) descriptions. We split the \(i\)th quality layer into \((N-m+i)\) equal parts and apply the \((N, k, N-k+1)\) Reed-Solomon code to get the contribution from the \(i\)th layer to each of the \(N\) descriptions. The contributions from each of the \(m\) quality levels are concatenated to form the \(N\) descriptions (see Figure 3). Thus, every description contains all \(m\) layers, and all \(N\) descriptions are equal in information content as intended. The combination of the progressively encoded bitstream and the family of Reed-Solomon codes enables us to attain quality level \(i\) provided that the number of erasures does not exceed \(m - i\), which results in the graceful degradation profile that we are after.

Now the question is how to optimally partition the resolution layers in Figure 2 and 3. In the next section, we will describe an algorithm that achieves this goal.

B. Optimal Allocation Algorithm

In this section, we propose a fast algorithm for choosing the quality profile \(d(k)\), that is optimal in the sense of maximizing the expected quality delivered at the receiver. We assume that our model is characterized by the transmission profile, \(q_i(N)\), where \(i = -1, \ldots, N-1\) denoting the probability that \(i+1\) out of \(N\) packets are delivered to the destination.

In this section we prove some key results for the case of a continuous R-D curve that can also be applied for the case of practical discrete R-D characteristics and would hold up to a convex hull approximation, but we do not present the extension due to lack of space. Since the quality is a one-to-one function \((D(r))\) of the rate \(r\), ascertaining the quality profile \(d(k)\) of order \(N\) corresponds to finding the rate partition \((R_0, \ldots, R_{N-1})\) of the bitstream (see Figure 2). Note that \(R_i \leq R_j\) for \(i < j\) from the embedding requirement of the source.

Fig. 3. FEC-based MD codes for a quality profile of order \(m\), i.e. having \(m\) quality levels or sections.
Define the expected distortion $ED$:

$$ED = q_1 \cdot E + \sum_{j=0}^{N-1} q_j \cdot D(R_j),$$

(1)

where $E$ equals the source variance, the distortion encountered when the source is represented by zero bits. When the codes as outlined in the above section are used, the total rate $R_t$ equals:

$$R_t = \frac{R_0}{1} N + \frac{(R_1 - R_0)}{2} N + \ldots + \frac{(R_{N-1} - R_{N-2})}{N} N$$

(2)

or equivalently,

$$R_t = \sum_{j=0}^{N-1} \alpha_j R_j$$

(3)

where

$$\alpha_j = \frac{N}{(j+1)(j+2)}, j = 0, \ldots, N-2, \text{ and } \alpha_{N-1} = 1$$

Problem Statement: Given the number of packets $N$, each packet of size $L$ (i.e. a total rate budget $R^*$ = $NL$), an embedded bitstream with rate-distortion curve $D(r)$, and the transmission profile $q_i$; Find $R$ that minimizes $ED$ subject to $R_t \leq R^*$ (the resource constraint) and

$$R_0 \leq R_1 \leq \ldots \leq R_{N-2} \leq R_{N-1}$$

(5)

We note that the R-D curve for any source, is theoretically a convex curve with the property that $\lambda(R_i) > \lambda(R_j)$ for $i < j$ where $\lambda(R)$ denotes the absolute value of the slope at point $(R_i, D(R_i))$. We observe that if constraint (5) were not present, the problem stated above would default to a standard bit allocation problem [17], generalized to include the notion of weights in the form of $\alpha_i$ and $q_i$. The optimal solution subject only to the resource constraint is easily found using the theory of Lagrange Multipliers and we briefly illustrate the procedure. Introducing the Lagrangian [18] for this problem, we get:

$$L_c(R_1, \ldots, R_{N-1}, \Lambda) = q_1 \cdot E + \sum_{j=0}^{N-1} q_j D(R_j) + \Lambda \left( \sum_{j=0}^{N-1} \alpha_j R_j - R_t \right)$$

(6)

At optimality, the partial derivative of the Lagrangian function with respect to $R_i, i = 0, \ldots, N-1$ and $\Lambda$ equals zero. This yields:

$$\frac{q_i}{\alpha_i} \cdot \frac{dD(R_i)}{dR_i} + \Lambda = 0, i = 0, \ldots, N-1.$$  

(7)

In other words, at optimality the respective slopes are in the proportion $\frac{q_i}{\alpha_i}$ Since $\frac{q_i}{\alpha_i} \cdot \frac{dD(R_i)}{dR_i}$ is a constant at optimality, it is clear that for monotonically decreasing $\frac{q_i}{\alpha_i}$, the absolute value of $\frac{dD(R_i)}{dR_i}$ has to be a monotonically decreasing sequence in $i$. Since the rate distortion curve is strictly convex, it follows that for this case, (5) is satisfied automatically. However, monotonicity of $\frac{q_i}{\alpha_i}$ cannot be guaranteed in general as we have no control over the channel state information $q_i$. Hence, in general the constraints of (5) cannot be ignored.

We now prove a key result that sheds insight into the nature or form of the optimal solution to the original problem that does not ignore (5) yet enables us to apply the method described above.

**Fact** If $\frac{\alpha_n}{q_n} \leq \frac{\alpha_{n+1}}{q_{n+1}}$, then in the optimal solution, $R_n = R_{n+1}$.

**Proof** Assume that there exists an optimal solution such that $R_n < R_{n+1}$. Then take away $\Delta R$ bits from $R_{n+1}$ and give $(\frac{\alpha_{n+1}}{q_{n+1}} \Delta R)$ bits to $R_n$ (so that the resource constraint and the rate constraints (5) are satisfied). Now, the net decrease in the $ED$ cost which is the difference between the decrease in cost in the $n$th term and the increase for $n+1$ th term is:

$$\text{Decrease} = \Delta R \alpha_{n+1} \frac{q_n}{\alpha_n} \lambda(R_n) - \frac{q_n+1}{\alpha_{n+1}} \lambda(R_{n+1})$$

(8)

Since $\frac{\alpha_n}{q_n} \leq \frac{\alpha_{n+1}}{q_{n+1}}$ and $\lambda(R_n) > \lambda(R_{n+1})$, the decrease is positive, which is a contradiction because if a solution with $R_n < R_{n+1}$ were optimal, we could still improve on the solution by successively decreasing $R_{n+1}$ and increasing $R_n$ while satisfying the rate constraints. Hence, $R_n = R_{n+1}$ in the optimal solution. Q.E.D.

The above result serves to characterize the form of the optimal solution based on the nature of the $\frac{\alpha_i}{q_i}$ profile. We make two key observations from the above result. Firstly, if $\frac{\alpha_n}{q_n} \leq \frac{\alpha_{n+1}}{q_{n+1}}$, the optimal solution to the original problem is the same as that to a reduced problem where $R_n = R_{n+1}$ is replaced by $R_n$ so that $\alpha'_n = \alpha_n + \alpha_{n+1}$ and $q'_n = q_n + q_{n+1}$. Also we see that if $\frac{\alpha_n}{q_n} \leq \frac{\alpha_{n+1}}{q_{n+1}}$ then $\frac{\alpha_n}{q_n} \leq \frac{\alpha_{n+1}}{q_{n+1}} \frac{\alpha_{n+1}}{q_{n+1}} = \frac{\alpha_{n+2}}{q_{n+2}}$. In other words, the above observation can be successively applied to a monotonically increasing or flat section in the $\frac{\alpha_i}{q_i}$ profile, and all the corresponding rate variables are equal in the optimal solution thus reducing the dimensionality of the problem.

Based on the above observations, we propose an $O(N)$ al-
Congestion Control

For the target application, we need the variation in the instantaneous rate to be smooth and small when the connection capacity is invariant, and we need to generate an accurate transmission profile that can be used by the source coder. At the same time, we need the congestion control algorithm to be TCP-friendly [20], robust, and react quickly to the onset of congestion. Congestion control algorithms that are deployed in the current Internet typically use the linear increase multiplicative decrease (LIMD) paradigm for both window-based [1] and rate-based congestion control [2], [3]. In this paper, we are interested in rate-based congestion control because this is better suited to multimedia applications. Briefly, LIMD periodically adapts the sending rate of a connection by gently increasing the rate by an additive constant \( \alpha \) upon observing no packet losses (in order to probe for additional bandwidth), and aggressively decreasing the sending rate by a multiplicative constant \( \beta \) upon observing packet losses (in order to alleviate congestion).

The trade-offs of LIMD are quite well-known: LIMD has been shown to be robust, and asymptotically converges to fairness [21]; on the other hand, it reacts identically and aggressively to all packet losses, which is appropriate behavior for losses caused by persistent congestion (i.e. when there is a reduction in the “fair share” of the network bandwidth for the connection) but inappropriate behavior for losses due to non-persistent congestion (e.g. losses induced by the sender probing for additional bandwidth, or random channel loss in wireless networks). As we mentioned above, LIMD throttles the sending rate by a multiplicative factor \( \beta \), which is typically set to 0.5. \( \beta \) must be set to a large enough value that the sender will throttle its rate aggressively in response to persistent congestion. On the other hand, this causes a large fluctuation in the instantaneous rate even in steady state, when losses are only induced by the linear probing mechanism of LIMD. Large rate fluctuations are undesirable for several reasons: (a) the sender must estimate the average sending rate accurately, which can be difficult in dynamic operating conditions, (b) the sender must buffer packets when application sending rate exceeds the instantaneous sending rate, potentially increasing end-to-end delay and buffer requirements, and (c) most important to our source coder design, it makes computing the transmission profile very difficult.

We present a congestion control algorithm called LIMD/H (LIMD with history) that augments the basic LIMD algorithm with an additional mechanism to differentiate the cause of packet losses and take adaptive action accordingly. The LIMD/H congestion control algorithm has the following two key features:

1. LIMD/H uses the history of packet losses in order to distinguish between persistent congestion and non-persistent congestion.
2. LIMD/H reacts gently to non-persistent congestion in order to keep the sending rate variation to a minimum in steady state, but reacts aggressively to the onset of the persistent congestion in order to prevent congestion collapse.

We present the LIMD/H algorithm in three parts. We first describe the framework for end-to-end congestion control, then describe the LIMD/H algorithm, and finally discuss some of its properties.

A. Framework for Congestion Control

In our framework, the evolution of the transmission rate occurs over discrete time periods called epochs. The sender assigns a monotonically increasing sequence number for each packet transmission. At the end of each epoch, the receiver computes the fraction of received packets over the range of packet sequence number during the epoch. The receiver then sends a congestion feedback to the sender containing the loss fraction \( 0 \leq f \leq 1 \). Upon receiving the loss feedback, the sender executes the LIMD/H congestion control algorithm to adjust its sending rate.

B. LIMD/H Congestion Control

Before describing the LIMD/H algorithm, we revisit the vanilla LIMD algorithm for reference. Let \( r \) denote the sending rate, \( f \) denote the loss fraction, \( \alpha \) denote the linear increase constant, and \( \beta \) denote the multiplicative decrease constant. Using LIMD:

- if \( f = 0 \), \( r \leftarrow r + \alpha \).
- if \( f > 0 \), \( r \leftarrow r \times (1 - \beta) \).

Ideally, losses that are not induced by persistent congestion (such as probe losses) should trigger a gentle throttling of the sending rate while losses that are induced by persistent congestion should trigger an aggressive throttling of the sending rate.
The fundamental problem is therefore to relate the rate throttling factor $\beta$ to the cause of packet loss. An intuitive and highly simplistic heuristic to predict persistent congestion is the following: if the packet losses observed during an epoch are caused by a persistent congestion but the sender does not throttle aggressively enough, then the loss will continue; on the other hand, if the packet loss is a probe loss, then so long as the sender throttles enough to account for the loss, there will be no loss in the next epoch.

Taking these factors into account, we propose the LIMD/H algorithm that keeps track of a very simple history parameter, $h$. The $h$ variable is initially set to 1, is doubled if there is a packet loss in an epoch, and reset to 1 if there is not. Thus, the $h$ variable is a very simple mechanism to capture the history of packet loss in the previous epochs. Based on the value of the $h$ variable, LIMD/H exercises a graded multiplicative decrease upon packet loss:

- If $f = 0$, $r \leftarrow r + \alpha$, and $h \leftarrow 1$.
- If $f > 0$, $r \leftarrow r \times (1 - \beta \times h)$, and $h \leftarrow 2h$.

In case of LIMD/H, we typically set the multiplicative decrease constant $\beta$ to be a small value (between 0.1 and 0.2), in order to achieve smooth variations of sending rate when the available bandwidth is invariant. LIMD/H thus throttles its transmission rate gently when there was no packet loss in the previous epoch, and progressively more aggressively when previous epochs have also experienced packet loss (see Figure 6). Of course, the impact of this approach is that if $\alpha$ is maintained at the same value as in LIMD, then the channel probe is as aggressive as before but the rate throttling on loss is less aggressive, thereby inducing more packet drops.

In order to reduce the packet drops and in order to design the algorithm to be TCP-friendly, we need to adjust the $\alpha$ as shown below. In effect, our approach is to trade-off the aggressiveness in the increase phase for gentle throttling in the decrease phase, while still allowing for an exponential increase in the throttling factor in order to account for sudden congestion and to prevent congestion collapse.

It turns out that the above approach is very simplistic and can be further significantly improved by maintaining a history of the “envelope” of the peaks of the transmission rate (i.e. the rates when losses occur), and conditioning $\beta$ on whether the loss occurs within an acceptable band of the envelope of the peaks (predicted as non-persistent congestion) or not (predicted as persistent congestion). Due to space constraints, we do not present the enhanced algorithm, which is reported in [22]. Nevertheless, the basic idea of our approach is to throttle the rate gently upon initially observing loss, and then throttling the rate more aggressively by exponentially increasing the throttling factor when we predict persistent congestion. It turns out that because of the resulting low rate fluctuations in steady state, we can improve both the estimate of the sending rate and the transmission profile.

C. Properties of LIMD/H

Property 1: If a connection experiences a packet loss probability of $p$, then the average rate in LIMD/H is upper bounded by $\frac{T \sqrt{2(1-\alpha) - 2\beta}}{p(1-\alpha)}$, where $\alpha$ is the increase constant, $\beta$ is the decrease constant in LIMD/H, and $T$ is the epoch time (this result follows from a simple extension of the standard TCP rate computation [20]). The expected variation of rate is between $(1-\beta) \frac{T \sqrt{2\alpha}}{p\beta(2-\beta)}$ and $\frac{T \sqrt{2\alpha}}{p\beta(2-\beta)}$.

Property 2: From the above result, LIMD/H is TCP-friendly if $\alpha = \frac{3\beta^2}{4\beta + 1}$. This yields $[\alpha, \beta]$ pairs of $[0.15, 0.1]$, $[0.33, 0.2]$, $[0.53, 0.3]$, $[1, 0.5]$, etc.

The trade-off is between the aggressive channel probing and gentle throttling. Note that simply choosing low values of $\alpha$ and $\beta$ in the standard LIMD algorithm is insufficient, because that causes very slow response to persistent congestion and leads to congestion collapse in highly dynamic environments. LIMD/H provides the fallback to aggressive throttling on persistent congestion while fluctuating around the average rate value at steady state.

We now briefly revisit the interaction between the transcoder and the congestion control modules. These two algorithms are very well matched: LIMD/H is able to provide a fairly accurate estimate of the connection capacity, and an aggregation of the transmission rate samples from the LIMD/H algorithms can be used to generate an accurate transmission profile.

IV. COORDINATION OF TRANSCODER AND CONGESTION CONTROL

In this section, we discuss the mechanism employed by the transport layer for the transmission profile calculation and its actual implementation. Figure 7 shows the interaction of the components presented in this paper, i.e. the source transcoder, LIMD/H congestion control algorithm, and the interaction layer between the two, in the end-to-end architecture of the system.

Among the input data to the MD-FEC block, Rate-Distortion characteristic $D(r)$ and the MR stream are user-level inputs, the transmission profile $q_i$ and the rate budget $R^*$ are connection level inputs. Thus, the remaining issues are:

1. How should the transmission profile and rate budget be computed?
2. How should the interaction between the transcoder and the transport be architected?

From the network’s point of view, it is easy to compute a set of $[p_i, r_i]$ pairs based on the transmission rate history from the congestion control algorithm, where $p_i$ is the probability of the achieving an effective transmission rate of $r_i$. Given this profile and the maximum allowable rate budget $R^*$, we can generate exactly how many packets can be transmitted in an epoch, and convert rate-based probabilities $p_i$ to packet probabilities $q_i$ required by the transcoder.

We first describe the interaction of the transport layer and source coder in 4 steps, and then detail the transmission profile generator algorithm:

1. When an application opens a socket, it notifies the kernel using the setsockopt option.
2. The congestion control module in the kernel queues the effective transmission rate into the kerneldq message queue. This message queue is a special queue that is accessible to both the kernel and the kerneld daemon.
3. When the kernel gets scheduled periodically, it dequeues the messages from the kernel dq message queue, and signals the appropriate process.

4. The runtime library then uses the effective transmission rate samples in order to update its transmission profile.

   Note that the kernel need not switch context or make an upcall to notify the application about the new transmission rate samples.

   We maintain a weighted running average of the frequency \( f_i(t) \) of occurrence of each rate sample \( r_i \), and generate the transmission profile by normalizing the frequencies.

   \[
   f_i(t) = \frac{\gamma f_i(t-1) + (1-\gamma)n_i(t)}{\sum_i f_i(t)}.
   \]

   where \( \gamma = 0.875 \) in order to maintain the long term history, and \( n_i(t) \) is the number of instances of \( r_i \) in the kernel dq during the \( t^{th} \) invocation of the transmission profile generator. Additionally, we estimate the rate budget \( R^* \) in the following way.

   First, we choose \( R^* \) to be the smallest rate value at which the aggregate probability of exceeding the rate is under 5%, i.e.

   \[
   R^*(t) \leftarrow \min_i \{ r_i \} \text{ s.t. } \sum_{j > i} p_j(t) < 0.05.
   \]

   Then we reduce the \( R^* \) by an amount that is proportional to the usual rate decrease in a steady state, i.e. \( k x \beta' x h \), where \( k = 0 \) is a tunable constant, which controls the conservativeness of the estimate, \( \beta' \) is the decrease factor, and \( h \) is the graded decrease factor in LIMD/H. The updated transmission profile and rate budget are passed onto the transcoder periodically which then adapts to the dynamic connection quality.

   To summarize, the focus of the interaction has been on the mechanism of sharing information between the transport and source layers. While our current mechanisms despite their simplistic nature perform well in the simulations, we propose to delve into the large body of related work on traffic modeling for future refinements of our approach.

V. SIMULATION RESULTS

In this section, we present simulation-based performance results of the proposed video transmission scheme. The ns-2 network simulator was used to implement the congestion control algorithms as a part of the transport layer protocol. For our simulations, we used a parameterized version of the “Football” video sequence as encoded by the state-of-the-art embedded video encoder, namely the 3-D SPIHT encoder [19]. The frame size is 352 x 240 pixels and each GOP comprises of 16 frames. The transmission speed chosen was 1 GOP per epoch, where each epoch has a duration equal to 650 msec, which corresponds to approximately 26.7 frames per second.

The reference scheme for our congestion control mechanism is LIMD. The reference systems for robust video encoding represent a gradual evolution from the dumb MR scheme to the proposed MD-FEC scheme. They are:

1. Multiresolution (MR) Source Encoder
2. Constant FEC or Equal Error Protection (EEP) and
3. Fixed Unequal Error Protection (FUEP).

MR source encoder corresponds to no robustness against packet loss. EEP is the case when different layers of the prioritized bit stream are assigned with the same level of error protection. EEP thus amounts to source unaware channel encoding since layers of different importance are encoded by identical protection. FUEP corresponds to the case when different layers of the prioritized bit stream are assigned unequal error protection that is fixed, and therefore does not adapt to the varying network conditions. An example of an FUEP system is the priority encoding transmission (PET) [23] scheme applied to encoding of MPEG video [24]. The performance and characteristics of the reference systems are summarized in Table I.

The performance of the video transmission systems was compared in terms of the measured delivered quality at the receiver (or peak signal-to-noise ratio (PSNR) in dB) and the smoothness of the quality variation over time perceived by the user. First we compare the effect of candidate congestion control algorithms. Then we provide performance results of the candidate robustness mechanisms in a simple network topology under conditions of random loss and sudden network capacity changes. The random loss model captures the commonly accepted connection model for a flow that shares a large network with a large number of other flows. The scenario where the network capacity suddenly changes illustrates the behavior of the system for networks with heavy dynamics or upon the occurrence of major network events such as the NASA pathfinder webcast. Finally we present the performance of the proposed video transmission system in a multi-hop network to show that its properties scale to larger network configurations. In order to illustrate the performance results, we present the PSNR (dB) on y-axis versus time (second) on x-axis curve in all the cases.

A. Comparison of the Congestion Control Schemes

We first compare the performance of candidate congestion control algorithms in an ideal network environment. The ideal network support for the transmission of a prioritized source consists of smart priority dropping switches in face of congestion. In this scenario, the source stream can be transmitted without introducing any robustness mechanisms since the network is endowed with the ability to discriminate and drop only the least priority packets.

The network topology for this experiment is shown in Figure 9 except that there was no CBR flow in this case. As a result, we observe that LIMD delivers a widely varying quality due to the large variations in the transmission rates, while LIMD/H delivers a more or less constant quality due to its tight hovering about the connection capacity (Figure 8). Owing to its suitability for transmission of multimedia applications and above mentioned features, henceforth, we consider only the LIMD/H for all our experiments unless explicitly mentioned otherwise.
<table>
<thead>
<tr>
<th></th>
<th>MR</th>
<th>EEP</th>
<th>FUEP</th>
<th>MD-FEC (proposed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Robustness</td>
<td>none</td>
<td>fixed FEC</td>
<td>fixed FEC</td>
<td>adaptive FEC</td>
</tr>
<tr>
<td>Priority notion</td>
<td>yes (first-k)</td>
<td>no</td>
<td>no</td>
<td>(any-k)</td>
</tr>
<tr>
<td>Quality profile</td>
<td>-</td>
<td>cliff</td>
<td>staircase</td>
<td>staircase</td>
</tr>
<tr>
<td>Overhead</td>
<td>none</td>
<td>pre-specified</td>
<td>pre-specified</td>
<td>adaptive</td>
</tr>
</tbody>
</table>

**Summary of performance**
- Jittery quality; highly susceptible to losses
- Two step quality; fails beyond the pre-specified correction capability
- Multi-step quality; multi-step quality; suffers during sudden network changes but adapts quickly to the new conditions

**TABLE I**
Comparison of different robustness schemes

---

**Fig. 8.** Effect of congestion control algorithm in the ideal video streaming scenario, i.e. MR with priority dropping. (a) LIMD, and (b) LIMD/H.

**Fig. 9.** A One-hop Network Topology

**B. Comparison of the Source Robustness Mechanisms**

In this section, we focus on the performance of various robustness mechanisms with LIMD/H congestion control algorithm in a simple network topology shown in Figure 9. In all the cases, we tested the performance of each system when there is random losses in the network channel, and when there is a sudden fair share reduction due to the introduction of 800 Kbps CBR on cbr → sink during 300 – 500 sec period.

The performance of the MR encoder is shown in Figures 10. As expected the MR encoder is not robust to even small variations in the sending rate employed by LIMD/H and delivers a widely fluctuating quality, the amplitude of the fluctuations being more than 5 dB (Figure 10.b before the introduction of CBR flow). In the situation of random loss, since there is a greater variation in the transmission rate owing to more frequent cutbacks, the situation is even worse.

Figure 11 shows the performance of the EEP scheme. In this case, the quality profile exhibits a cliff effect. We observe that for the given simulation environment, protection against 10 losses is not even enough to provide immunity against probe losses (see Figure 11.b before the introduction of the CBR flow). This simulation brings out the fundamental tradeoff between the redundancy of the codes and the peak quality delivered: greater redundancy implies more robustness but lower source quality delivered.

The FUEP scheme assigns prespecified protection to prespecified source resolution layers. For instance, in [24], the Priority Encoding Transmission scheme [23] was used for MPEG encoding. Here the I-frames were encoded with priority 60%, P-frames with priority 85% and B-frames with priority 95%. As in [24], in FUEP, we partitioned the source into three layers (the relative rates of the layers being 70%, 20%, and 10%, similar to typical relative rates for I, P, and B frames in MPEG streams). Figures 13 and 12 compare the performance of the FUEP scheme and the MD-FEC scheme. We observe that for random losses, MD-FEC in general delivers quality better than FUEP by about 2 dB. For the case of sudden capacity reduction, MD-FEC quickly adapts to the new available capacity whereas

---

8Protecting a resolution layer with priority $\alpha$% means that the receiver can recover it from $\alpha$% of the encoded packets. In channel coding jargon, $\alpha$ is referred to as the code rate.
FUEP takes time to adapt to the new environment because it does not adaptively protect the high priority data according to the transmission profile.

At this point, the performance improvement of MD-FEC over FUEP scheme may not seem significant. However, the fundamental difference between MD-FEC and FUEP is that MD-FEC is an adaptive scheme which adapts the level of protection according to the network dynamics whereas FUEP is not. To verify the advantage of adaptivity of MD-FEC, we considered two FUEPs with different protection level; one with large code rate and the other with small code rate. In particular, we consider the FUEP source with the following protection level: (a) a low redundancy source with channel code rates of 70%, 80%, and 95% for high/medium/low priority packets (and hence is optimized for networks with small packet loss), and (b) a high redundancy source with channel code rates of 40%, 70%, and 95% (and hence is optimized for networks with high packet loss). We compare the performance of these FUEPs with MD-FEC in network environments changing the packet loss probability (0.01% – 1% loss). Figure 14 summarizes the result.

In this simulation, we observe the following: (a) LIMD congestion control may result in very jittery delivered quality at the receiver when the data is not well protected (case (a) and (b)), (b) FUEP (even with LIMD/H) may deliver either very jittery quality (case (d)) when the loss protection is not sufficient, or very low PSNR (case (e)) when the loss protection is unnecessarily high. However, MD-FEC scheme adapt to the appropriate protection level using the transmission profile, i.e. when loss rate is small it adapts to be like the FUEP with small overhead (case (g) and (c)), and when loss rate is high it adapts to be like the FUEP with large overhead (case (h) and (f)). In other words, the adaptive nature of MD-FEC enables it to operate with desirable FEC overhead, whereas the non-adaptivity of FUEP makes it break when the pre-specified protection level does not match with the current network condition.

C. Performance in a Large Network

In this section, we tested the performance of the robustness mechanisms in a complex multi-hop network topology. Since we have observed that FUEP and MD-FEC scheme provides best performance in the simple network topology, we only present the results with FUEP and MD-FEC due to the space constraints. Figure 15 shows the topology of the network. There are 2 video streams ($v_{s1}$ → $v_{r1}$, $v_{s2}$ → $v_{r2}$) and 10 TCP connections ($tcp_i$ → $sink_j$) in the network. The link capacity and the one-way delay is annotated on the graph. All the links without annotation were set to 40 Mbps and 20 msec.

Figure 16 shows the performance results of the two video streams when encoded by the FUEP and MD-FEC methods. We tested FUEP with various code rates and presented the best result.

We observe that in general the MD-FEC scheme delivers a higher end-to-end quality than the FUEP scheme. Although there are occasional spikes in the delivered quality due to the instantaneous deviation of transmission profile computation, in general the delivered quality is less jittery than that delivered by the FUEP scheme. In the case of FUEP, the quality profile is very sensitive to the choice of protection level and a bad choice
of code rate either results in low amplitude or highly jittery quality. To summarize, MD-FEC scheme responds reasonably well to varying network dynamics and is more adaptive than either of the reference systems.

VI. RELATED WORK

Prior work on transmission of prioritized data over error/erasure channels includes [14], [15] and has focused largely on image transmission. Priority Encoding Transmission [23] is an algorithm that assigns Forward Error Correction (FEC) codes, according to priorities specified by the user, to message fragments (also specified by the user) transmitted over packet networks. However, the algorithm does not specify how to assign priorities or how to fragment the message; these tasks are left to the user.

In [14], RCPC (Rate Compatible Punctured Convolutional) codes were used to protect images compressed with the progressive image encoder 2D-SPIHT [10] over binary symmetric noisy channels. The choice of rates for the channel codes was empirical. In [15], the issue of image transmission over packet erasure channels was considered. Unlike the PET scheme, they proposed an algorithm that assigns FECs to various message fragments and chooses the message fragments. The algorithm is based on greedy iterative descent techniques and is suboptimal.

Signal Processing based methods for introducing robustness into the source include the classical approaches to the MD problem that are based on quantizers [11], correlating transforms [12] and newer approaches based on wavelet polyphase decompositions [13]. Most of these approaches suffer from the issue of implementation complexity and the fact that they are difficult to generalize to more than two descriptions except for in [13], [25].

Park et al. proposed the framework of an adaptive FEC (AFEC) for delay-sensitive real-time traffic over network [26]. AFEC adopts a simple control algorithm which decreases the degree of the redundancy when the network is well-behaved and increases the redundancy otherwise. However, they do not specify how to distribute FEC in order to maximize the expected receiver quality in the context of unequal error protection context.

Another issue pertinent to this work is the end-to-end congestion control of multimedia connections. RAP (Rate Adaptation Protocol) [2] adopts the rate-based LIMD rate adaptation paradigm in order to provide fairness among the multimedia streams as well as the TCP-friendliness. The issue has been further investigated in [27]. The main idea of this work is to combine the coarse grain adaptation (by adding or deleting layers) and the fine grain rate-based LIMD adaptation using buffering at the receiver, which enables a smooth playback.

SCP (Streaming Control Protocol) [3] tries to adjust its sending rate smoothly according to its bandwidth and round-trip time estimation. However, the increase algorithm for bandwidth probing and the decrease algorithm upon packet loss or timeout is essentially LIMD. Since SCP is sensitive to packet loss (as LIMD is), the bandwidth and round-trip estimation must be accurate, which may be very difficult to achieve in a dynamic network environment.

Much of the prior work has either focused on the aspects of source coding with simplistic channel models or on the transport layer with simplistic source models. Our goal is to address the combined end-to-end problem with synergistic coordination between the source coder and the transport layer.

VII. CONCLUSIONS AND FUTURE WORK

In recent years, multiresolution coding has become a very popular paradigm for image/video source coding. However, the current network model of the Internet is not well suited for the transmission of MR bitstreams because most routers typically do not consider application-specified priorities of the packets when making the decision to drop packets during congestion. Furthermore, the predominantly used linear increase multiplicative decrease congestion control paradigm results in large rate variations at the source and makes it difficult to accurately compute the transmission profile.

In order to address these issues, we presented two components: (a) the MD-FEC transcoding algorithm that converts MR streams to non-prioritized MD streams that are able to better tolerate variations in the frequency and relative position of packet loss, and (b) the LIMD/H congestion control algorithm that reduces the fluctuation in the sending rate when available connection capacity is invariant while at the same time being responsive to congestion. Finally, we presented an efficient mechanism to coordinate the congestion control and source coding algorithms closely in order to best use the varying network resources. Our preliminary performance evaluation has shown that our approach is viable and worth more detailed study.

One key component in our approach that needs more work is the estimation of the transmission profile. Our architecture admits transmission profile generators as pluggable modules, and we are currently exploring the use of more sophisticated traffic models to estimate this profile.

REFERENCES


