Design Issues for Layered Quality-Adaptive Internet Video Playback

Reza Rejaie[1] and Amy Reibman[2]

[1] AT&T Labs- Research, Menlo Park, CA. reza@research.att.com
[2] AT&T Labs- Research, Florham Park, NJ. amy@research.att.com

Abstract. The design of efficient unicast Internet video playback applications requires proper integration of encoding techniques with transport mechanisms. Because of the mutual dependency between the encoding technique and the transport mechanism, design of such applications has proven to be a challenging problem. This paper presents an architecture which allows the joint design of a *transport-aware* video encoder with an *encoding-aware* transport. We argue that layered encoding provides maximum flexibility for efficient transport of video streams over the Internet. We describe how off-line layered encoding techniques can achieve robustness against imprecise knowledge about channel behavior (*i.e.*, bandwidth and loss rate) while maximizing efficiency for a given transport mechanism. Then, we present our prototyped client-server architecture, and describe key components of the transport mechanism and their design issues. Finally, we describe how encoding-specific information is utilized by transport mechanisms for efficient delivery of stored layered video despite variations in channel behavior.

1 Introduction

The design of efficient unicast Internet video playback applications requires proper integration of encoding techniques with transport mechanisms. Because of the mutual dependency between the encoding technique and the transport mechanism, design of such applications has proven to be a challenging problem.

Encoding techniques typically assume specific channel behavior (*i.e.*, loss rate and bandwidth) and then the encoder is designed to maximize compression efficiency for expected channel bandwidth while including a sufficient amount of redundancy to cope with the expected loss rate. If channel behavior diverges from expected behavior, quality of delivered stream would be lower than expected. The shared nature of Internet resources implies that behavior of Internet connections could substantially vary with unpredictable changes in co-existing traffic during the course of a session. This requires all Internet transport mechanisms to incorporate some type of *congestion control* mechanism (*e.g.*, [1], [2]). Thus to pipeline a pre-encoded stream through a congestion controlled connection, video playback applications should be able to efficiently operate over the expected range of connection behaviors. This means that video playback applications should be both *quality adaptive* to cope with long-term variations in bandwidth and *loss resilient* to be robust against range of potential loss rates.

In this paper, we argue that layered encoding is the most promising approach to Internet video playback. Layered encoders structure video data into layers based on importance, such that lower layers are more important than higher layers. This structured representation allows transport mechanisms to accommodate variations in both available bandwidth and anticipated loss rate, thus enabling simpler joint designs of encoder and transport mechanisms. First, transport mechanisms can easily match the compressed rate with the average available bandwidth by adjusting delivered quality. Second, transport mechanisms can repair missing pieces of different layers in a prioritized fashion over different time scales. Over short timescales (*e.g.*, a few round trip times (RTT)), lost packets from one layer can be recovered before any lost packet of higher layers. This allows transport mechanisms to control the observed loss rate for each layer. This is crucial because neither total channel loss rate nor its distribution across layers are known a priori and can be expected to change during transmission. Over long timescales (*e.g.*, minutes), the server can *patch* quality of the delivered stream by transmitting pieces of a layer that were not delivered during the initial transmission. Over even longer timescales, the server can send extra layers, to improve quality of a previously-transmitted stream without being constrained by the available bandwidth between client and server. This allows adjustment of quality for a cached stream at a proxy[3].

We have prototyped a client-server architecture (Figure 1) for playback of layered video over the Internet. The congestion control (CC) module determines available bandwidth (BW) based on the network feedback (*e.g.*, client's acknowledgment). The Loss recovery (LR) module utilizes a portion of available bandwidth (BW_{lr}) to repair some recent packet losses such that the observed loss rate remains below the tolerable rate for the given encoding scheme. The server only observes the remaining losses that have not been repaired, and uses the remaining portion of bandwidth (BW_{qa}) to perform Quality adaptation (QA)[4] by adjusting delivered quality (*i.e.*, number of transmitting layers). The QA and LR mechanisms each depend on parameters of the specific encoding and are tightly coupled. The collective performance of the QA and LR mechanisms determine the perceived quality of the video playback. A key component of the architecture is a Bandwidth Allocator (BA) that divides total available bandwidth between the QA and LR modules using information that depends on the specific encoding and on client status.

This paper describes our ongoing work to integrate transport-aware encoding with encoding-aware transport for Internet video playback. We consider a coupled design, in which the encoder and transport are each designed given knowledge of the expected behavior of the other. For transport-aware encoding, we present the main design choices and trade-offs for layered encoding, and describe how the encoding schemes can be customized based on available knowledge regarding employed transport mechanism and regarding expected channel rate and loss rate. For the design of encoding-aware transport mechanisms, we focus on design strategies for the BA, QA and LR modules that are directly affected by details of the deployed layered encoding scheme.

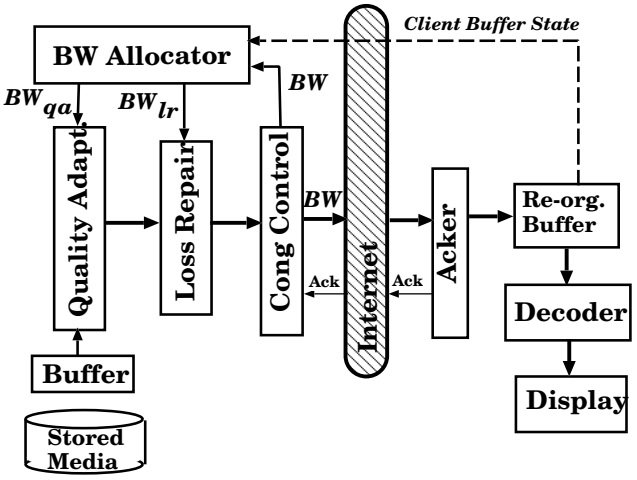

Fig. 1. Client-server Architecture

The rest of this paper is organized as follows. In section 2, we review some of the related work. Because of the natural ordering between encoding and transport, we consider transport-aware encoding using a layered video encoder in section 3. Then in section 4, we address various components of an encoding-aware transport mechanism and examine their design tradeoff. This includes Bandwidth allocation (section 4.1), Loss Recovery (section 4.2) and Quality adaptation (section 4.3). Section 5 concludes the paper and addresses some of our future plans.

2 Related Work

Traditionally, video encoders have been designed for transport over fixed-rate channels (with fixed and known channel bandwidth) with few if any losses. This creates a bit stream that will have poor quality if transported over a network with different bandwidth or loss rate. Either a higher loss rate or a lower bandwidth would produce potentially significant visual artifacts that propagate with time. For transport over networks, the encoder design should change to be cognizant of the fact that the bit stream may have to deal with varying channel bandwidth and non-zero loss rate. Over the years, several classes of solutions have been proposed for encoding and transmitting video streams over the network as follows:

– *One-layer Encoding*: In one-layer video encodings (*e.g.*, MPEG-1 and MPEG-2 Main Profile), the trade-off between compression and resilience to errors is achieved by judiciously including Intra-blocks, which do not use temporal prediction. The choice of which blocks to code as I-blocks for a given sequence can be optimized if channel loss rate is known a priori [5].

Transport mechanisms can not gracefully adjust quality of a one-layer pre-encoded stream to available channel bandwidth. A common solution is to deploy an *Encoding-specific packet dropping algorithm* [6]. In these algorithms, the server discards packets that contain lower-priority information (*e.g.*, drops B frames of MPEG streams) to match transmission rate with channel bandwidth. The range of channel rates over which these algorithms are useful is usually limited, and the delivered quality may be noticeably degraded. Both these effects are content- and encoding specific.

- *Multiple description coders*: Another approach is to use a multiple description (MD) video encoder [7]. MD coders are typically more robust to uncertainties in the channel loss rate at the time of encoding. However, they still require similar network support to adapt to varying channel bandwidth.
- *Multiple Encodings*: One alternative to cope with variations in channel behavior is to maintain a few versions of each stream, each encoded for different network conditions. Then the server can switch between different encodings in response to changes in network conditions. Limitations of this approach are the inability to improve quality of an already-transmitted portion of the stream, and the inability of such a system to quickly respond to sharp decreases in available bandwidth.
- *Layered Encodings*: Hierarchical encoding organizes compressed media in a layered fashion based on its importance *i.e.*, layer i is more important than all higher layers and less important than all lower layers. If the lower more important layers are received, a base quality video can be displayed, and if the higher less important layers are also received, better quality video can be displayed. The layered structure of encoded stream allows the server to effectively cope with uncertain channel behavior.

In summary, layered encoding has three advantages: First, layered video allows easy and *effective rate-matching*. The server can match the bandwidth of delivered stream with the available network bandwidth by changing the number of layers that are transmitted. This relaxes the need for knowing exact channel bandwidth at the time of encoding, thus it helps to decouple transport and encoding design.

Second, layered video allows *unequal error protection* to be applied during transport, with stronger error correction applied to the more important layers. This can be used effectively even if the expected loss rate is not known at the time of compression. Thus, even though one-layer video and the most important layer of layered video are equally susceptible to losses, discrepancies between the actual loss rate and the one assumed at the time of encoding can be accommodated by the transport mechanism.

Third, layered encoding allows the server to improve quality of an already-transmitted portion of the stream. We call this *quality patching*. In essence, layered structure allows the server to deliver different portions of various layers in any arbitrary order, *i.e.*, reshape the stream for delivery through uncertain channel. Thus quality of the delivered stream can be smoothed out over various timescales. Figure 2 illustrates flexibilities of layered encoding to reshape the

stream for transmission through the network. The server adjusts the number of layers when there are long-term changes in available bandwidth. Total loss rate is randomly distributed across the active layers. However, the server can prioritize loss recovery by retransmitting losses of layer i before layer j for any $i < j$. When extra bandwidth becomes available at time t_3, the server can either add the fourth layer or alternatively transmit five missing segments of the layer 2 (between t_1 and t_2). This shows how a layered encoded stream can be reshaped a stream for delivery through the network.

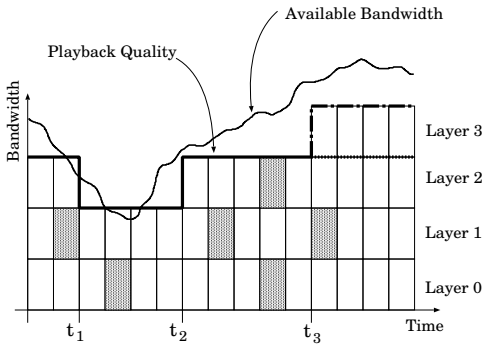

Fig. 2. Flexibility of Layered Encoding

3 Transport-Aware Layered Encoding

In this section, we consider how best to create a video bit stream to be stored at the media server in Figure 1. We argued earlier that a layered (or in this paper, equivalently a scalable) encoder will produce a better bit stream than a one-layer encoder if the values of channel bandwidth and loss rate are not known at the time of encoding. Despite these advantages inherent in layered video, good system performance requires careful design. We begin with some background on layered video encoders, and then show how incorporating at the time of encoding as much information as possible about the channel bandwidth and loss rate can generate the best bit stream for storage. We also discuss the information that needs to be shared between encoder and transport to improve quality of delivered video.

3.1 Layered Coding Background

There are two basic methods to create a layered video bit stream: 3-dimensional wavelets [8, 9] and adding layering to a traditional DCT-based video coder with inter-frame temporal prediction [10, 11]. The former has drawbacks of quite poor compression performance in sequences with complex motions, and poor video

quality when temporal sub-bands are lost due to motion blurring. For these reasons, we focus on the family of layered DCT-based temporally-predictive video coders in this paper.

Among this family of coders, there has been much research and several standardized coders. In these coders, P-frames are predicted from previous frames and the DCT coefficients are coded in a layered fashion. Several important aspects of these coders are the followings:

- how the DCT coefficients are partitioned between each of the layers (*i.e.*, is partitioning done in the frequency domain, the spatial domain, or the quality or SNR domain),
- whether enhancement-layer information uses temporal prediction at all, and if so whether lower-layer base information is used to predict the higher-layer enhancement information,
- how much bandwidth is allocated for each layer,
- how much of that bandwidth is redundancy in each layer,
- and how many layers are created.

These decisions all affect the compression efficiency of the layered coder, and also affect the robustness against packet loss. Thus, knowledge of the expected bandwidth and loss rates at the time of transport can have a large impact on the best choice of codec design.

While most standardized layered encoders produce two or at most a few layers [11, 10], recent interest in having many layers has lead to a standard for MPEG-4 Finely Granular Scalability (FGS) [12]. The bit stream structure of MPEG-4 FGS makes it highly robust and flexible to changes in available bandwidth. However, such robustness comes with a significant penalty to the efficiency of the compression and hence to the video quality as the rate increases. More recent, not yet standardized approaches to layered encoding markedly improve the compression efficiency without too much sacrifice to the robustness [13].

The design of a layered encoder is based on underlying assumptions regarding the nature of the transport and channel. Specifically, a layered coder assumes that the transport mechanism will initially attempt to send the more important information in the available bandwidth, and that the loss recovery will be applied to the more important parts before the less important parts. This implicitly requires buffering at the client and in the transport. However, the exact bandwidth and loss rate experienced at the time of transport is typically not known at the time of encoding, and further assumptions must be made. In general, these assumptions have been implicit in the design of the layered video encoder. Here we make them explicit. We consider first the bandwidth, then consider the loss rate.

3.2 Incorporating Bandwidth Knowledge

The more knowledge available at the time of encoding regarding the expected range of operating bandwidths, the better. We focus on how knowledge of the

available bandwidth impacts the prediction strategy of the encoder. If the available bandwidth is known to range between R_{min} and R_{max}, three different prediction strategies are useful depending on the expected behavior of the bandwidth within this range.

First, consider the case where the bandwidth is nearly always close to R_{max}. Then the best design would be to rely heavily on prediction for the enhancement layers so as to improve compression efficiency. (This is essentially the strategy of a one-layer encoder.) Such a design will suffer small degradation when bandwidth dips slightly below R_{max}, but in such an environment the probability of larger degradations is small.

Second, consider the case where the bandwidth is usually near R_{min} although we'd like better quality when the rate is higher. Then the best design would be to use temporal prediction only for the base layer with rate R_{min}, and to use no temporal prediction for the higher layers. (This is the strategy used by MPEG-4 FGS.) Such a design is very robust to variations in bandwidth in $[R_{min}, R_{max}]$, but is also not very efficient at rates near R_{max}.

Third, suppose we have little knowledge of the bandwidth other than it lies in the range $[R_{min}, R_{max}]$. In this case, the best algorithm is to judiciously choose the prediction strategy for each macro block in the video, so as to balance both compression efficiency (at rates near R_{max}) and robustness (at rates near R_{min}). Figure 3 illustrates these concepts for the sequence *Hall monitor* using the scalable video coder in [13]. This coder has the flexibility of using three different methods of prediction for the different layers, which allows it to mimic both a one-layer video coder and the MPEG-4 FGS video coder through an appropriate restriction of the prediction methods. The results in Figure 3 are obtained by creating a single bit stream for each illustrated curve, and successively discarding enhancement-layer bit-planes and decoding the remainder. The x-axis shows the decoded bit-rate, and the y-axis shows the PSNR of the resulting decoder reconstruction as bit-planes are discarded. Also shown is the performance of the one-loop encoder with no loss (top dotted line). This provides an upper bound on the performance of the scalable coders.

In this figure, the curve labeled "FGS" uses the prediction strategy of the MPEG-4 FGS coder which is optimal if the available bandwidth is usually near R_{min}. The curve labeled "one-layer with loss" uses a one-layer prediction strategy, which is optimal if the available bandwidth is usually near R_{max}. The curve labeled "drift-controlled" is our coder [13] optimized to provide good performance across the range of rates.

The FGS coder performs poorly at the higher rates. The one-layer decoder with drift suffers a 2.6-4.3 dB degradation at the lowest bit-rate, compared to the drift-free FGS decoder. Relative to the FGS coder, our proposed coder suffers about 1.3-1.4 dB performance degradation at the lowest bit-rate, but significantly outperforms it elsewhere. Our coder loses some efficiency at the highest rates compared to the one-layer coder, but has noticeably less drift as bit-planes are discarded.

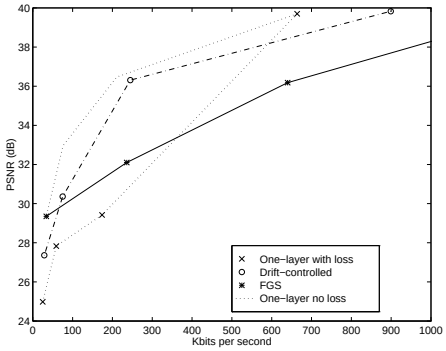

Fig. 3. Effect of Bandwidth on PSNR

Table 1 shows the PSNR averaged across different channel rates, assuming a uniform distribution of rates between the smallest and the largest rate of the one-loop encoder. The coder optimized for the range of channel rates outperforms the other coders by 0.8-2.1 dB when there is only one I-frame.

seq.	One-loop	Proposed	FGS	upper
Hall	33.14	35.25	33.11	36.77
Fore	33.33	34.13	33.09	35.03

Table 1. PSNR assuming uniform distribution.

3.3 Incorporating Loss Rate Knowledge

Next, we consider the effect of incorporating knowledge of the expected loss rate into the video encoder design. We distinguish loss rate from lowered bandwidth because losses will randomly affect the entire frame for a given layer, while lowered bandwidth can be targeted specifically toward less important parts of the bit stream within a frame for a given layer.

In the Figure 3, we showed how the choice of prediction structure used by the layered encoder is influenced by the expected bandwidth. Similarly, the expected loss rate affects the best prediction structure. Figure 4 shows the performance of a two-layer H.263 encoder under random losses in each layer. Two different prediction strategies are used for the enhancement layer. The base layer is compressed with 64 Kbps, and each enhancement layer is compressed with 128 Kbps. The curve labeled "Enh 128" corresponds to an enhancement layer which is predicted only from the base layer, while the curve labeled "Enh 128p" corresponds to an enhancement layer that also uses prediction from previous enhancement-

layer pictures. Performance for "Enh 128" and "Enh 128p" assume the base layer is completely received.

For the base layer, performance degrades gradually as the loss rate increases from 0.1% to 1%, but as the loss rate continues to increase, the performance degrades significantly. Thus it will be important for the transport mechanism to keep the residual losses (after loss recovery) below 1% for the base layer.

The more aggressive prediction strategy of "Enh 128p" has visually better performance for low loss rates, but for loss rates greater than about 5%, performs significantly worse than the less efficient prediction strategy of "Enh 128". (Note that the coder in [13] should not have such significant degradations. This coder will be used in the final paper to illustrate this figure.) In both cases, performance with 100% loss of enhancement layer is identical to the base-only performance. If the more aggressive prediction strategy is used, then it will be important for the transport to keep the loss rate for the enhancement layer below 10%, while if the less efficient prediction strategy is used, the loss rate for the enhancement layer is less critical. However, performance will be significantly degraded in the case of few losses, or if additional layers are added beyond this first enhancement layer.

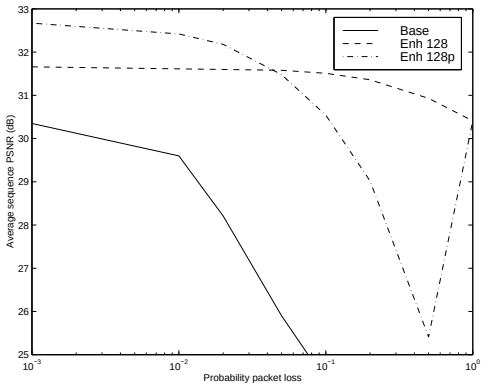

Fig. 4. Effect of Loss on PSNR

3.4 Information Provided to Transport

To enable the best design of an encoding-aware transport, the encoder should provide some meta-information to the transport mechanism. This meta-information includes two pieces of information:

1. The bandwidth-quality trade-off (*e.g.*, Figure 3). More specifically, the encoder conveys the bandwidth (or consumption rate) of each layer (*i.e.*, C_0, C_1, ..., C_N), and the improvement in quality caused by each layer (*i.e.*, Q_1, Q_2, ..., Q_N).

2. The loss rate and quality trade-off (*e.g.*, Figure 4). In some situation (such as for the base-layer or the more aggressive enhancement-layer prediction strategy in Figure 4) the loss rate vs. quality meta-information can be reduced to simple thresholds regarding the maximum tolerable loss rate for each layer (*i.e.*, L_{max0}, L_{max1}, ..., L_{maxN}).

In general, this meta-information will be encoding and even content specific. Ideally, it should also include temporal variations, indicating the trade-offs for different scenes within a sequence, or for each frame in a sequence. However, in practice, it might be necessary to use static information (for the entire sequence) or generic information (for a class of sequences, like "head-and-shoulders").

The interface between the transport and the encoder is completely characterized by the loss rate and the bandwidth dedicated to video information. Therefore, the above information will be sufficient to design effective encoding-aware transport mechanisms.

4 Encoding-Aware Transport

In this section, we illustrate how a transport mechanism for layered encoded streams can leverage encoding-specific information to improve quality of delivered stream. First, we present our client-server architecture to identify main components of the architecture and their associated design issues. Then, we explore design space of the main components to show how encoding-specific information can be used to customized the design. Our goal is to clarify key tradeoffs in the design of a transport mechanism for layered video and demonstrate how they can use information about encoded streams to improve delivered quality over the best-effort Internet.

Figure 5 depicts our client-server architecture for delivery of Internet video playback (Figure 1) with more details. As we mentioned earlier, the architecture has four key components: 1) Congestion Control (CC) is a network-specific mechanism that determines available bandwidth (BW) and loss rate (L) of the network connection. Available bandwidth and loss rate are periodically reported to the Bandwidth Allocation (BA) module. The BA module uses encoding-specific information to properly allocate the available bandwidth between the Loss Recovery (LR) and the Quality Adaptation (QA) modules. The LR module utilizes allocated portion of available bandwidth (BW_{lr}) to retransmit required ratio of recent losses. The remaining portion of available bandwidth (BW_{qa}) is used by the QA module to properly determine quality of delivered stream (*i.e.*, number of transmitting layers).

All the streams are layered encoded and stored. Thus, different layers can be sent with different rate (bw_0, bw_1, ..., bw_n). The server multiplexes all active layers along with retransmitted packets into a single congestion controlled flow. The client demultiplexes different layers and rebuilds individual layers in virtually separate buffers. Each layer's buffer is drained by the decoder with a rate equal to its consumption rate (*i.e.*, C_0, C_1, ..., C_n). The client reports its playout time in each ACK packet. This allows the server to estimate the client's buffer

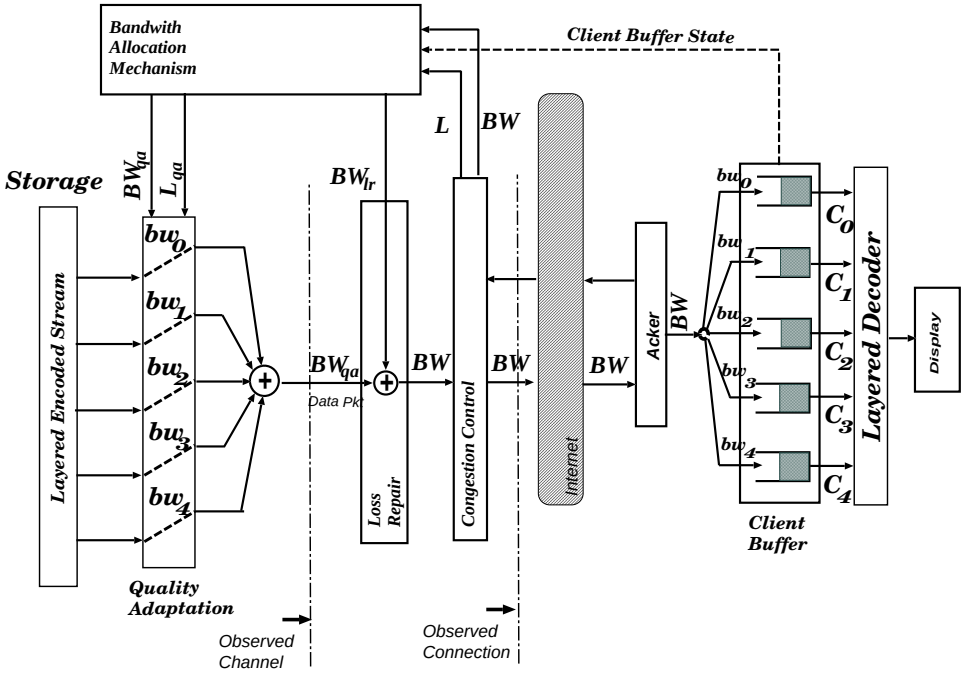

Fig. 5. Client-Server Architecture for Streaming of Layered Video

state, *i.e.*, the amount of buffered data for each layer. Client buffering is used to absorb short-term variations in bandwidth without changing the delivered quality.

The main goal of the transport mechanism is to map the actual connection bandwidth and loss rate into the range of acceptable channel behavior expected by the encoding mechanism. Layered encoding provides two levels of freedom for the transport mechanism to achieve this goal: 1) by changing number of layers, the transport mechanism can adjust the required channel bandwidth for delivery of the stream, and 2) by allocating a portion of available channel bandwidth to loss repair, the transport mechanism can reduce the observed loss rate. Therefore, there are three key issues in design of a transport mechanism for layered encoded streams that can be tailored for a given encoded stream to improve delivered quality:

- *Bandwidth Allocation strategy*: How should the transport mechanism allocate total connection bandwidth between LR and QA modules?
- *Loss Repair strategy*: How should the loss repair bandwidth (BW_{lr}) be shared among transmitting layers?
- *Quality Adaptation strategy*: How should the server adjust the quality of delivered stream (*i.e.*, the number of layers) as available channel bandwidth (BW_{qa}) changes?

Since congestion control is a network-specific mechanism, its design should not be substantially affected by application requirements. Therefore, we do not discuss design issues of the CC mechanism. The main challenge is that the behavior of a network connection (BW and L) is not known a priori, and even worse it could substantially change during the course of a connection. Thus, the server should adaptively change its behavior as the connection behavior varies.

For the rest of this section, we provide insight in each one of the above three strategies in design of transport mechanism for layered encoded stream and demonstrate how the transport mechanism can benefit from encoding-specific information. We assume that the encoding-specific meta-data (described in section 3.4) are available for each layered encoded stream.

4.1 Bandwidth Allocation

BA module shifts the connection loss rate (L) into the range of acceptable loss rate by allocating the required amount of connection bandwidth for loss repair. Therefore, the application will observe a channel with a lower loss rate (L_{qa}) at the cost of lower channel bandwidth. We need to drive a function that presents the tradeoff between the channel bandwidth (BW_{qa}) and the channel loss rate (L_{qa}). Given the connection bandwidth (BW) and the connection loss rate (L), the total rate of delivered bits is equal to $BW(1-L)$. Therefore, the ratio of delivered bits for the channel is $\frac{BW(1-L)}{BW_{qa}}$. Thus, we can calculate the channel loss rate as follows:

$$L_{qa} = 1 - \frac{BW(1-L)}{BW_{qa}} \quad (1), \quad \text{where } BW_{qa} \leq BW \text{ and } BW_{qa} \geq BW(1-L)$$

Equation (1) presents L_{qa} as a function of BW_{qa} for a given connection (*i.e.*, BW, L). Figure 6 depicts this function for different set of BW and L values. Each line in Figure 6 represents possible channel behaviors for a given network connection as the BA module trades BW_{qa} with L_{qa}. For example, point A represents a connection with 1000 Kbps bandwidth and 40% loss rate. To reduce the channel loss rate down to 33% (*i.e.*, shifting point A to point B), the BA module should allocate 100 Kbps of connection bandwidth for loss repair, whereas reducing the loss rate down to 14% (*i.e.*, shifting point A to point C) requires the BA module to allocate 300 Kbps of the connection bandwidth for loss repair.

Figure 6 clearly demonstrates how the BA strategy can be customized for a given encoded streams using the information provided by the encoder. Given the bandwidth of various layers (*i.e.*, C_0, C_1, ...) and the per-layer maximum tolerable loss rates (*i.e.*, L_{max0}, L_{max1}, ...), to find the maximum number of layers (n) that can be delivered through a network connection (BW, L), the following two conditions should be satisfied:

$$\frac{\sum_{i=0}^{n} L_{max\,i}}{n} \geq L_{qa} \quad (2), \qquad \sum_{i=0}^{n} C_i \leq BW_{qa} \quad (3)$$

The first condition examines that the average loss rate for n active layers is less than the channel loss rate, whereas the second condition ensures that channel bandwidth is sufficient for delivery of n layers. Given the values of BW, L and $n = N$ (where N is maximum number of layers), the BA module should use equation (1) to search for a channel loss rate that satisfies equation (2). If such a channel loss rate can be accommodated while the corresponding channel bandwidth satisfies equation (3), then n layers can be delivered and total required bandwidth for loss repair is (BW_{lr} = BW - BW_{qa}). Otherwise, the BA module decreases n by one and repeats this process.

The BA module continuously monitors the connection behavior to determine the required bandwidth for loss repair such that the channel behavior always satisfies the conditions (equation 2 and 3) for the number of active layers. If the connection loss rate increases or the connection bandwidth decreases such that these conditions can no longer be satisfied, the BA module signals the QA module to drop the top layer. This decrease n which in turn presents a new set of conditions to the transport mechanism.

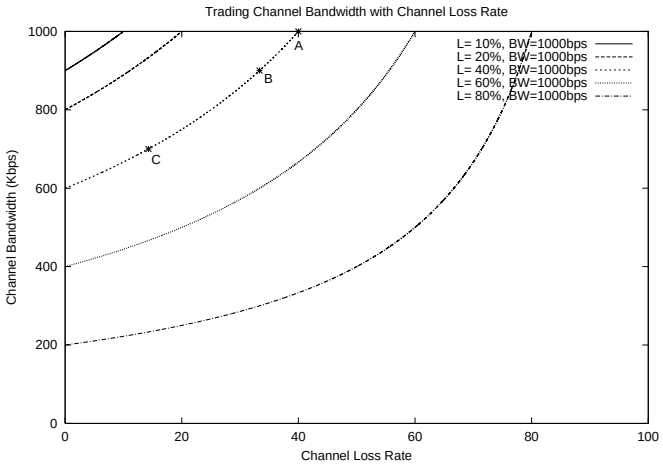

Fig. 6. Trading the channel bandwidth with the channel loss rate

In summary, the BA module determines the bandwidth share for the QA and LR modules. This allows us to separate the design of the Loss recovery mechanism from the Quality adaptation mechanism in spite the fact that their collective performance determines delivered quality.

4.2 Loss Recovery

The loss repair module should micro-manage the total allocated bandwidth for loss repair (BW_{lr}) among the active layers such that the loss rate observed by

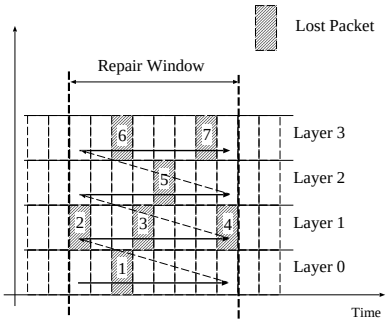

Fig. 7. Sliding Window approach to prioritized Retransmission

each layer remains below its maximum tolerable threshold (*i.e.*, L_{max0}, L_{max1}, ..., L_{maxN}). Since all layers are multiplexed into a single unicast session at the server, the distribution of total loss rate across active layers is seemingly random and could change in time. Thus, the bandwidth requirement for loss recovery of various layers can randomly change in time even if total loss rate remains constant. We assume a retransmission-based loss recovery since 1) retransmission is feasible for playback applications with sufficient client buffering, 2) retransmission is more efficient (*i.e.*, requires less bandwidth) than other repair schemes such as FEC, and 3) retransmission allows fine-grained bandwidth allocation among active layers [1].

Since the importance of a layer monotonically decreases with its layer number, loss repair should be performed in a prioritized fashion, *i.e.*, losses of layer j should be repaired before losses of higher layers and after losses of lower layers. However, a prioritized approach to loss repair should ensure that the total repair bandwidth is properly shared among active layers and that each retransmitted packet is delivered before its playout time. To achieve this, we deploy a sliding-window approach to loss repair as shown in Figure 7. At any point of time, the server examines a recent window of transmitted packets across all layers. Losses of active layers are retransmitted in the order of their importance such that the loss rate observed by each layer remains below its maximum tolerable loss rate (*i.e.*, L_{maxi}). Figure 7 shows order of retransmission within a window for each layer and across all layers.

The repair window should always be a few round-trip-times (RTT) ahead of the playout time to provide sufficient time for retransmission. Therefore, the repair window slides with playout time. If the BA module properly estimates the required bandwidth for loss repair, all losses can be repaired. However, if the allocated bandwidth for loss repair is not sufficient to recover all the losses within a window, this approach repairs the maximum number of more important losses. The length of the repair window should be chosen properly. A short window

[1] Although we only discuss retransmission-based loss repair, the basic idea can be applied to other post-encoding loss repair mechanisms such as unequal FEC.

cannot cope with a sudden decrease in bandwidth, whereas a long window could result in the late arrival of retransmitted packets for higher layers. In summary, the sliding window approach to prioritized loss repair, 1) uses maximum per-layer tolerable loss rates for an encoded stream to improve its performance, and 2) adaptively changes distribution of total repair bandwidth among active layers.

4.3 Quality Adaptation

The QA mechanism is a strategy that adds and drops layers to match the quality of the delivered stream (*i.e.*, number of transmitting layers) to the channel bandwidth (BW_{qa}). When the channel bandwidth is higher than the consumption rate for active layers, the server can use the extra bandwidth and send active layers with a higher rate (*i.e.*, $bw_i > C_i$) to fill up the client buffer. The buffered data at the client can be used to absorb a short-term decrease in bandwidth without dropping any layers. Figure 8 illustrates the filling and draining phases of the client buffers where three layers are delivered. If the total amount of buffered data during a draining phase is not sufficient to absorb the decrease in bandwidth, the QA module is forced to drop a layer. Consequently, the more data that is buffered at the client during a filling phase, the bigger the reductions that can be absorbed. The amount of buffered data at the client side is determined by the strategy of adding layers.

Figure 9 compares two adding strategies. During a filling phase, the QA strategy adds a new layer after a specific amount of data is buffered. If the required amount of buffered data is small, the QA mechanism aggressively adds a new layer whenever the channel bandwidth slightly increases. In this case, any small decrease in bandwidth could result in dropping the top layer because of small amount of buffering. Alternatively, the QA mechanism can conservatively add a new layer only after a large amount of data is buffered.

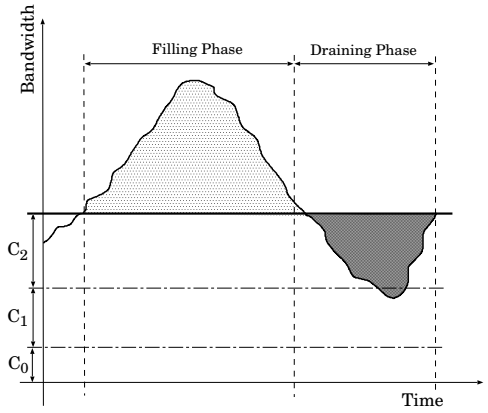

Fig. 8. Filling and Draining phases for Quality adaptation

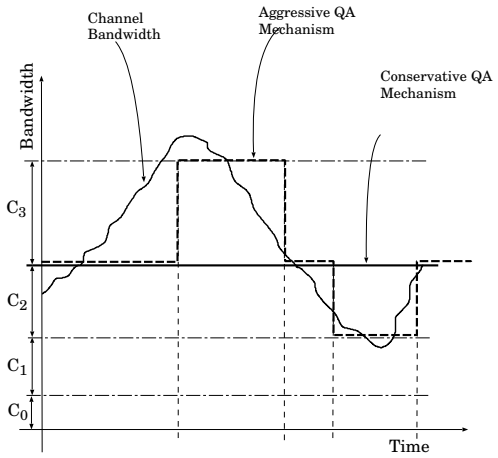

Fig. 9. Effect of adding strategy on quality changes

More buffered data allows the server to maintain the newly added layer for a longer period of time despite major drops in bandwidth. Figure 10 shows an aggressive and a conservative adding strategies in action[4]. The congestion-controlled bandwidth is shown with a saw tooth line in both graphs. This experiment clearly illustrates the coupling between the adding and dropping strategies. *The more conservative the adding strategy, the longer a new layer can be kept, resulting in fewer quality changes, and vice versa.*

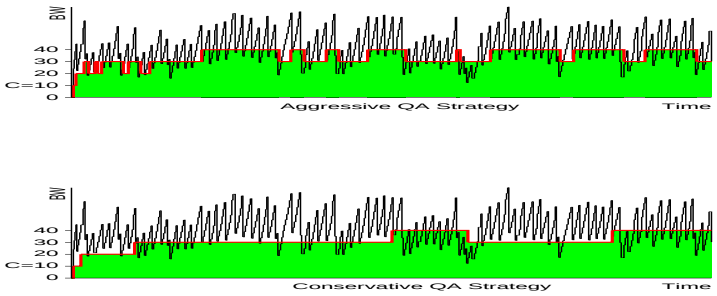

Fig. 10. Aggressive vs Conservative QA Strategies

The QA mechanism should be customized for the particular layered stream being transmitted. To explain this, we need to examine two basic tradeoffs in the design of an add and drop strategy.

– *How much data should be buffered before adding a new layer?*
This should be chosen such that normal oscillation in channel bandwidth in the steady state does not trigger either adding or dropping a layer. Figure 8 clearly shows that the required amount of buffered data to survive a drop in bandwidth directly depends on the total consumption rate of active n layers (*i.e.*, $\sum_{i=0}^{n} C_i$).
– *How should the buffered data be distributed among active layers?*
Since the streams are layered encoded, buffered data should be properly distributed across all active layers in order to effectively absorb variations in bandwidth. During a draining phase, buffered data for layer i cannot be drained faster than its consumption rate (C_i). Therefore, buffered data for layer i can not compensate more than C_i bps. Figure 11 illustrates this restriction. To avoid dropping a layer, the total draining rate of buffering layers (*e.g.*, $C_2 + C_1$) should always be higher than the deficit in channel bandwidth (BW_{def}). More specifically, during a draining phase the following conditions must be satisfied:

$$BW_{def} = \sum_{i=0}^{n} C_i \text{ - } BW_{qa}, \qquad BW_{def} \leq \sum_{i \in BufLayers} C_i$$

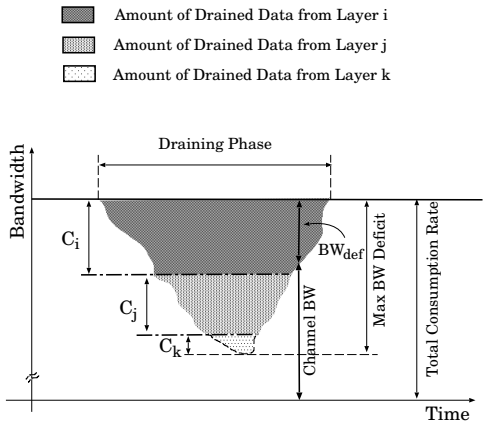

Fig. 11. Sample distribution of buffered data among buffering layers

5 Conclusion and Future Work

In this paper, we presented a joint design of encoder and transport mechanisms for playback of layered quality-adaptive video over the Internet. The main challenge is that the Internet does not support QoS. At the time of encoding, the channel bandwidth and loss rate are not known and they could significantly

change during the course of a session. Therefore, traditional encoding approaches that assume static channel behavior will result in poor quality.

We argued that layered video is the most flexible solution because 1) it can be efficiently delivered over a range of channel behavior, and 2) it provides sufficient flexibility for the transport mechanism to effectively reshape the stream for delivery through the variable channel. However, to maximize quality of delivered video, the encoding should become transport-aware and the transport mechanism should become encoding-aware. Toward this end, we described several issues in the design of layered encoding mechanisms and explained how the expected range of channel bandwidth and loss rate information can be incorporated into the encoding mechanism. Furthermore, the encoding mechanism provides encoding-specific meta-information. The transport mechanism uses this information to bridge the gap between the expected range of channel behavior by the encoder and the actual connection behavior. More specifically, we provided insight on how the main components of the transport mechanism, particularly Bandwidth Allocation, Loss Repair and Quality Adaptation, can leverage encoding-specific meta-information to improve the delivered quality of layered video despite unpredictable changes in channel behavior.

Finally, we plan to conduct extensive experiments over the Internet to evaluate the overall performance of our client-server architecture. This will allow us to identify those scenarios in which our architecture can not properly cope with changes in connection behavior. Some of these problems can be addressed by the encoding mechanism through appropriate provisioning, whereas others require further tuning or modification of the transport mechanism. Our experiments should provide deeper insight about channel behavior that may suggest refinement of the layered encoding mechanism. We also plan to examine interactions among three key components of the transport mechanism, as well as implications of congestion control algorithm on other components of transport mechanism.

References

1. R. Rejaie, M. Handley, and D. Estrin, "RAP: An end-to-end rate-based congestion control mechanism for realtime streams in the internet," *Proc. IEEE Infocom*, Mar. 1999.
2. S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-based congestion control for unicast applications," *Proc. ACM SIGCOMM*, Sept. 2000.
3. R. Rejaie, H. Yu, M. Handley, and D. Estrin, "Multimedia proxy caching mechanism for quality adaptive streaming applications in the internet," in *Proc. IEEE Infocom*, Tel-Aviv, Israel, Mar. 2000.
4. R. Rejaie, M. Handley, and D. Estrin, "Quality adaptation for congestion controlled playback video over the Internet," *Proc. ACM SIGCOMM*, Sept. 1999.
5. R. Zhang, S. L. Regunathan, and K. Rose, "Video coding with optimal inter/intra-mode switching for packet loss resilience," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 6, pp. 966–976, June 2000.
6. Kang-Won Lee, Rohit Puri, Tae eun Kim, Kannan Ramchandran, and Vaduvur Bharghavan, "An integrated source coding and congestion control framework for

video streaming in the internet," in *Proc. IEEE Infocom*, Tel-Aviv, Israel, Mar. 2000.

7. A. R. Reibman, H. Jafarkhani, Y. Wang, and M. Orchard, "Multiple description coding for video using motion compensated prediction," in *International Conference on Image Processing*, Oct. 1999.
8. D. Taubman and A. Zakhor, "Multirate 3-D subband coding of video," *IEEE Trans. Image Processing*, vol. 3, no. 5, pp. 572–588, Sept. 1994.
9. S. McCanne, V. Jacobson, and M. Vetterli, "Receiver-driven layered multicast," *Proc. ACM SIGCOMM*, Aug. 1996.
10. R. Aravind, M. R. Civanlar, and A. R. Reibman, "Packet loss resilience of mpeg-2 scalable video coding algorithms," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 5, pp. 426–435, Oct. 1996.
11. G. Cote, B. Erol, M. Gallant, and F. Kossentini, "H.263+: Video coding at low bit rates," *IEEE Transaction on Circuit and Systems for Video Technology*, vol. 8, no. 7, Nov. 1998.
12. H. Radha et. al., "Fine-granular-scalable video for packet networks," in *Packet Video Workshop*, New York NY, Apr. 1999.
13. A. R. Reibman and L. Bottou, "Managing drift in a DCT-based scalable video encoder," in *IEEE Data Compression Conf.*, Mar. 2001.