Optimal Quality Adaptation for MPEG-4 Fine-Grained Scalable Video

Taehyun Kim and Mostafa H. Ammar

Abstract— Dynamic behavior of the Internet's transmission resources makes it difficult to provide perceptually good quality of streaming video. MPEG-4 Fine-Grained Scalable coding is proposed to deal with this problem by distributing the data in enhancement layers over a wide range of bit rates. However, encoded video also exhibits significant data rate variability to provide a consistent quality video. We are, therefore, faced with the problem of trying to accommodate the mismatch between the available bandwidth variability and the encoded video variability. In this paper, we investigate quality adaptation of the layered VBR video generated by MPEG-4 FGS. Our goal is to develop a quality adaptation scheme that maximizes perceptual video quality through minimizing quality variation while at the same time increasing the usage of available bandwidth. We develop an optimal adaptation scheme and an online heuristic based on whether the network conditions are known *a priori*. Experimental results show that the online heuristic as well as the optimal adaptation algorithm provide consistent video quality when used over both TFRC and TCP.

I. INTRODUCTION

In recent years, the number of popular web sites serving multimedia content has led to the growth of streaming video applications. One of the challenging aspects of carrying video over today's Internet is the fact, as was identified in [15], that the Internet's transmission resources exhibit variability at multiple time scales, and the available bandwidth fluctuates over a broad range because of the wide distribution of packet loss burst duration, changes in bottleneck capacity, and multiple time scale queuing-time variation. This dynamic behavior of the Internet makes it difficult to provide perceptually good quality of streaming video.

Small time scale variability can be accommodated by utilizing a receiver buffer, where a few video frames can be prefetched before they are displayed. However, it is difficult to accommodate large time variability using the receiver buffer because of the buffer size limitation. Large time scale variability is generally accommodated using scalable (or layered) video encoding [2], [16]. In this approach, a source video is encoded into a base layer and one or more enhancement layers. The base layer can be decoded independently and provides the minimum video quality. Enhancement layers can be decoded cumulatively and contribute to the improvement of video quality. MPEG-4 FGS (Fine-Grained Scalability) provides efficient scalable video coding in the framework of a standard

encoding technique. This scheme can be highly adaptable to the Internet's bandwidth fluctuation by distributing the data in enhancement layers over a wide range of bit rates. However, the encoded video can exhibit significant rate variability if the encoding is targeting consistent perceptual quality (this is a result of the compression techniques, which are based on the use of DCT, quantization, motion compensation, and entropy coding). We are, therefore, faced with the problem of trying to accommodate the mismatch caused by both the available bandwidth variability and the encoded video variability.

In this paper, we investigate adaptive video streaming techniques for layered VBR MPEG-4 video. The goal is the development of an *optimal* algorithm that minimizes the quality variability while at the same time maximizing the utilization of the variable network bandwidth. Our starting point is the traditional rate adaptation scheme for layered video delivery, such that video layers are added and dropped as the available bandwidth changes. However, it is generally agreed that significant quality fluctuation caused by frequent adding and dropping layers may be annoying and degrade the perceptual quality of video.

The paper is organized as follows. We first briefly review related work in Section II. Section III describes the layering model based on MPEG-4 FGS. In Section IV, we consider the quality adaptation algorithms and the question of optimizing quality adaptation. Section V reports on experimental results we have conducted. The paper is concluded in Section VI. A companion web site provides a demonstration of the basic results of this work [3].

II. RELATED WORK

Accommodating data rate variability using a receiver buffer has been widely deployed. Authors in [18] proposed an optimal rate smoothing algorithm based on the work-ahead smoothing technique for non-layered VBR video which achieves minimum variability of transmission rate. Combined with guaranteed service or RCBR service [6], it was shown that network utilization can be increased significantly. However, rate smoothing is not useful for a best effort network, since the Internet does not provide any information about the bandwidth evolution in advance. Also, a smooth data rate does not always guarantee a smooth quality for VBR video.

Coarse grained adaptation of layered video has been discussed in the context of multicasting to accommodate the heterogeneity of receivers [8], [10], [11]. The number of layers subscribed by a receiver is dynamically varying, since a receiver adjusts the video quality based on the network

condition: subscribing to as many layers as possible when the available bandwidth is large, and dropping layers when the available bandwidth is small. However, frequent adding and dropping of layers can incur significant quality variability which leads to the degradation of perceptual video quality.

Quality adaptation algorithms based on layered video for a unicast environment have been also proposed in [4], [13], [17]. The algorithm in [4] transforms the quality adaptation problem into a shortest path problem to minimize variability. The algorithm in [13] tries to maximize the perceptual video quality by using bidirectional optimum layer selection. The algorithm in [17] accommodates the short-term rate variability caused by a TCP friendly congestion control mechanism. However, these algorithms assume all layers are CBR encoded, hence they do not maximize perceptual quality of VBR video.

The problem of layered VBR video streaming was addressed in [19]. The authors model the available bandwidth as a stochastic process and propose an optimal bandwidth allocation scheme among base and enhancement layers. However, as the objective of this scheme is to minimize the loss probability in the base and enhancement layers, it may incur significant quality variation. Since this scheme has the closest (but not the same) objective as we do, we compare it with our algorithms and investigate the impact on perceived visual quality.

III. RATE VARIABILITY IN MPEG-4 FGS

The growing need for an efficient scalable video coding motivated the development of MPEG-4 FGS, which was established in the Amendment on Streaming Video Profile of the MPEG-4 standard [2]. Several features have been employed to satisfy this demand [9], [16]: 1) it accommodates a wide range of data rate variability by distributing enhancement layers over a wide range of bit rates, 2) it provides an efficient coding based on bit-plane coding which is more efficient than run-level coding, and 3) it separates the FGS layer from the motion compensation stage to eliminate drift in the enhancement layer.

In this section, we describe the MPEG-4 FGS framework, and we evaluate its rate variability of the encoded video using a software codec in [16]. The FGS framework consists of a base layer and one or two enhancement layer. Two types of enhancement layers are defined for hybrid temporal-SNR scalability in MPEG-4 FGS: 1) *SNR FGS layer* contributes to enhancing video quality by adding DCT coefficients with a reduced quantization step size, which leads to highly accurate DCT coefficients and high quality video. 2) *Temporal FGS layer* is designed to improve temporal resolution by providing a higher frame rate and smooth motion.

The base layer is generated by DCT, ME/MC (Motion Estimation and Motion Compensation), and entropy coding. To provide consistent quality, it is necessary to fix the quantization step size of an encoder. However, the constant quantization step size inherently leads to data rate variability. The FGS layer (also called the SNR FGS layer) is created by DCT and entropy coding of the residual image of the base layer, hence it does not provide constant bit rate if constant quantization step size is employed. The FGST layer (also called the Temporal

FGS layer) is generated by motion compensated residual frames, which are encoded in the same manner as the SNR FGS layer. Therefore, all layers of encoded video will exhibit data rate variability. Fig. 1 shows the data rate of an encoded 150-second scene from the movie: *A river runs through it* using a GOP size of 12. The video sequence is generated by a software MPEG-4 FGS codec in [16]. We can observe that all layers show significant rate variability. The variability spans as much as 7.4kBytes in the base layer, 33.6kBytes in the FGS layer, and 29.9kbytes in the FGST layer.

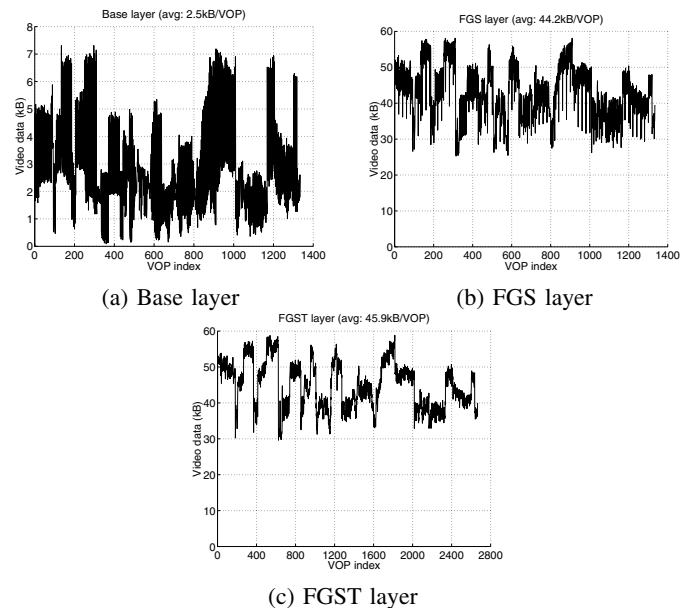(a) Base layer   (b) FGS layer

(c) FGST layer

Fig. 1. Data rate variability of encoded video. The video sequence is generated from *A river runs through it* using an MPEG-4 FGS codec.

We need to investigate the relative importance of the two types of enhancement layer to achieve optimal layering. Fig. 2 shows two possible implementations of a hybrid temporal-SNR scalability structure. The structure in Fig. 2 (a) places emphasis on the FGS layer to improve video quality. The structure in Fig. 2 (b) increases temporal resolution before improving video quality.

Fig. 3 shows decoded video quality measured in PSNR (Peak Signal-to-Noise Ratio) for the first 300 VOPs (Video Object Plane) of original video. Fig. 3 (a) shows decoded video quality when only a base layer is used. Since the base layer has low quality and low frame rates, PSNR is about 45dB and the number of VOPs is only 100. Fig. 3 (b) shows the result when the base and FGS layers are decoded together. Since the FGS layer provides an improvement of image quality, PSNR is significantly improved by more than 20dB. However, the frame rate is still low (one third of the full frame rate). The full frame rate can be achieved when the FGST layer is decoded together. Fig. 3 (c) shows the result when the base and the FGST layers are decoded. The frame rate is improved by three times compared with Fig. 3 (a) and (b). However, it does not provide a consistent quality of video, and the decoded video alternates between high quality and low quality. We can
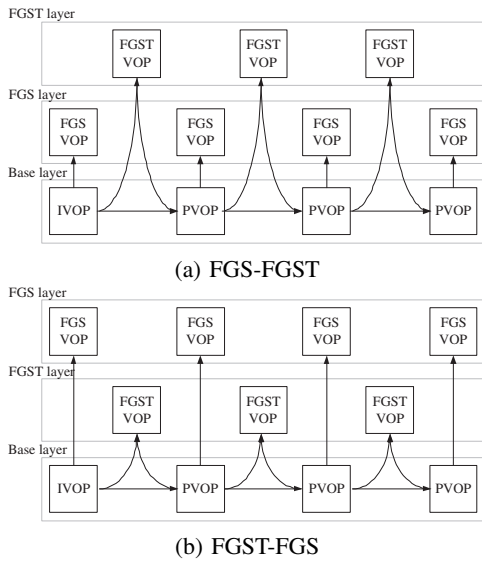
(a) FGS-FGST

(b) FGST-FGS

Fig. 2. Structures of hybrid scalability. Structure (a) improves video quality first and structure (b) increases temporal resolution first.

(a) Base layer only

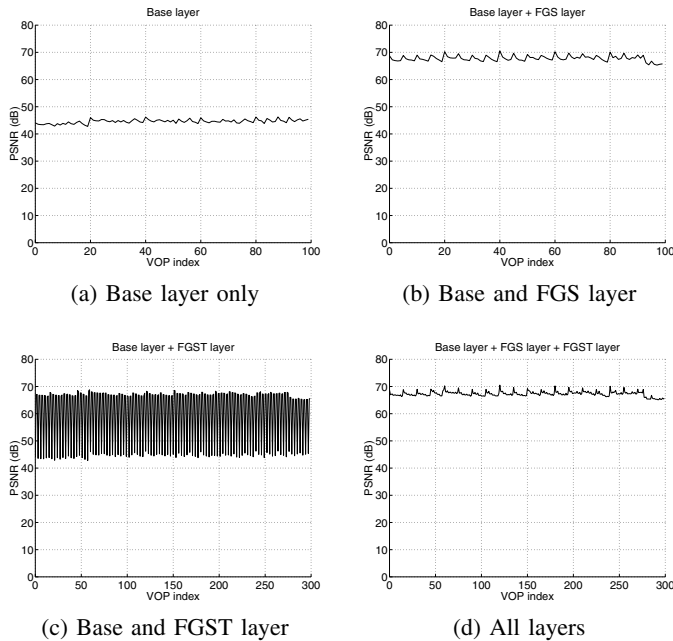(b) Base and FGS layer

(c) Base and FGST layer

(d) All layers

Fig. 3. Video quality measured in PSNR. Video quality when the base layer and the FGS layer are decoded together is shown in (b), but temporal resolution is not increased. Video quality of the base layer and the FGST layer is shown in (c), but it exhibits significant quality variability.

observe that a video of inconsistent quality is perceptually annoying, even if it has a high frame rate. Therefore, we place more importance to the FGS layer than the FGST layer. Fig. 3 (d) shows the result when all layers are decoded together. In this case, decoded video exhibits both high quality (67.3dB on average) and smooth motion (full frame rate). Readers are encouraged to verify this observation by downloading the decoded videos from [3].

IV. QUALITY ADAPTATION ALGORITHMS

In this section, we present quality adaptation algorithms for MPEG-4 FGS video. *Quality adaptation* is defined by a mechanism that adds and drops layers based on the available network bandwidth while maximizing the perceptual video quality (primarily we are interested in consistent "long runs" of the same quality video, this will be formalized in Section V). We assume that a sender maintains layered VBR video consisting of three layers: the base layer as layer 1, the FGS layer as layer 2, and the FGST layer as layer 3. A receiver is assumed to have some amount of buffering capacity which allows it to prefetch unplayed video. The available bandwidth may exhibit multiple time scale variability.

A. Composed Algorithm

As a baseline algorithm, we consider a straightforward approach for quality adaptation by combining existing algorithms. The quality smoothing algorithm proposed in [13] accomplishes the maximum reduction of quality variability for layered CBR video using bidirectional layer selection. In order to apply this algorithm to layered VBR video, we need to insure that the data rate of the encoded video is sufficiently smoothed to exhibit nearly constant bit rate. We adopt the rate smoothing algorithm presented in [18] for this purpose, since this algorithm enables a sender to transmit a piecewise CBR sequence by using the work-ahead smoothing technique. We compose a quality adaptation algorithm from the two algorithms.

We divide the receiver buffer into two portions: one used for rate smoothing and the other for quality smoothing. We first use the *FindOptimalSchedule* function in [18] to compute optimally smoothed transmission rate for each layer using rate smoothing buffer. Next, we apply the *MaxAvgRun* function in [13] to the smoothed rate video for maximizing average run length using quality smoothing buffer. The composed algorithm is given in Fig. 4, where L is the number of layers and N is the number of VOPs in a layer. However, as the algorithm in [13] assumes the size of a frame in a layer is normalized to 1, we need to modify it: replacing 1) $\theta = 1$ by $\theta = x_i[k]$ and 2) $S_k^i = 1$ by $S_k^i = x_i[k]$, where θ is a temporary variable determining a feasible sequence, $x_i[k]$ is the size of VOP k, and S_k^i is a feasible sequence of layer i.

```
1: procedure ComposedAlgorithm (r[], x_i[], b_r, b_q)
2:      for i = 1 to L do
3:          FindOptimalSchedule (x_i[], b_r)
4:      enddo
5:      Initialization: r_1[k] = r[k], where k ∈ {1, ..., N}
6:      for i = 1 to L do
7:          (r_{i+1}[], s_i[]) = MaxAvgRun(r_i[], x_i[], b_q)
8:      enddo
9: endprocedure
```

Fig. 4. Composed algorithm. A piecewise CBR stream is generated by the *FindOptimalSchedule* function, where b_r stands for the rate smoothing buffer size. Next the *MaxAvgRun* function is applied to achieve consistent quality, where b_q specifies the quality smoothing buffer size.

B. Optimal Quality Adaptation

We now develop a more sophisticated algorithm targeting optimal quality adaptation for MPEG4 FGS VBR video.

1) Framework: In formulating optimal quality adaptation, we consider a discrete time model. Let $x_i[k]$ be the VOP size of the ith layer at time k, b_i be the receiver buffer size for storing unplayed ith layer video, and $r_i[k]$ be the available bandwidth, where $i = 1, \ldots, L$ and $k = 1, \ldots, N$. Note that the available bandwidth $r_i[k]$ is the residual bandwidth after accommodating layers $1, 2, \ldots, i-1$. Hence the condition of $r_i[k] > 0$ implies $r_j[k] > 0$, where $j = 1, \ldots, i-1$. Conversely $r_i[k] = 0$ implies $r_{j'}[k] = 0$, where $j' = i + 1, \ldots, L$. $X_i[k]$ represents the cumulative data requirement in the ith layer defined by $X_i[k] = \sum_{j=1}^{k} x_i[j]$. The cumulative capacity, $C_i[k]$, is defined to quantify transmission resources. The cumulative capacity of the ith layer is determined by two constraints: the receiver buffer size and the available network bandwidth, such that

$$C_i[k] = \min(S_i[k-1] + b_i, \ C_i[k-1] + r_i[k]),$$

where $S_i[k]$ is the cumulative selected data defined by $\sum_{j=1}^{k} s_i[j]$, and $s_i[j]$ is the selected data defined by

$$s_i[j] = \begin{cases} x_i[j], & \text{if VOP } j \text{ is selected to be transmitted} \\ 0, & \text{otherwise} \end{cases}.$$

Note that $s_i[k]$ represents the transmission schedule for the ith layer: If $s_i[k]$ is not zero, a VOP of the ith layer of time k is transmitted to a receiver. On the other hand, if $s_i[k]$ is zero, it is not transmitted to the receiver. Especially, $s_i[k]$ is said to be *feasible* if the amount of cumulative selected data does not exceeds cumulative capacity, i.e., $S_i[k] = \sum_{j=1}^{k} s_i[j] \leq C_i[k]$.
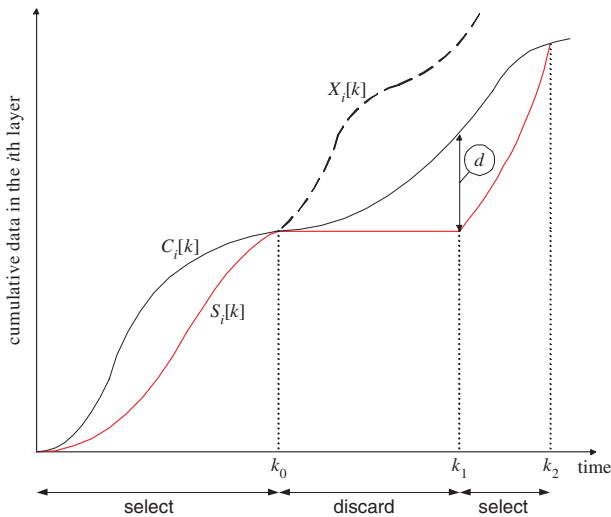

Fig. 5. Framework of quality adaptation. A receiver makes transitions between the select and discard state based on transmission resources.

Fig. 5 illustrates the framework of optimal quality adaptation. We assume that there are enough transmission resources in the beginning stage, such that a sender can transmit a video stream without discontinuity and therefore $S_i[k] = X_i[k]$, $k = 1, \ldots, k_0 - 1$. Once transmission resources are exhausted

at k_0, a receiver cannot display the video stream during the interval of (k_0, k_1), but the sender allows the receiver to prefetch the next selected VOPs starting from k_1. During (k_1, k_2), the receiver does not only display buffered video but also prefetch VOPs. Once the transmission resources are exhausted again at k_2, no VOPs can be transmitted and the receiver begins to prefetch the next selected VOPs. Although the quality adaptation framework in Fig. 5 seems similar to the rate adaptation mechanism in [18], there is a significant difference: The constraint of rate adaptation is determined by the receiver buffer size and the source video rate, whereas the main constraint of quality adaptation is transmission resources.

We model the quality adaptation mechanism by a two-state machine as shown in Fig. 6. In each layer, a VOP in the `select` state is transmitted and one in the `discard` state is dropped. Each state is determined by the amount of transmission resources. Since we cannot send more data than can be accommodated by transmission resources, the cumulative capacity is an upper bound of the cumulative transmission schedule of the ith layer; A sender transmits a video stream without discontinuity as long as the cumulative capacity is greater than the cumulative selected data of the encoded video. Hence, our task is to determine optimal threshold that maximizes perceptual video quality. The prefetch points (e.g., k_1 in Fig. 5) are determined by the threshold.
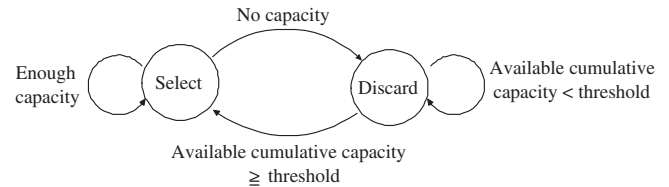

Fig. 6. State transition diagram specifying the quality adaptation mechanism.

2) Select State: In the `select` state, a VOP can be transmitted to a receiver. To achieve maximum perceptual quality, it is necessary to reduce quality variability, such that a sender transmits as many consecutive VOPs as possible. Hence, if the sender enters the `select` state, it does not leave the state until transmission resources are exhausted. Once transmission resources are not enough to accommodate the encoded video, the sender enters the `discard` state.

To accommodate multiple layers, we employ a conservative transmission policy: If there is available transmission resources, video data is transmitted as soon as possible from the base layer to the highest enhancement layer. The residual cumulative capacity of lower layers can be used to accommodate higher layers. In this scenario, lower layers are protected better than higher layers, since lower layers are buffered before higher layers. This makes it more likely to deliver lower layers than higher layers. Note that, if a VOP of the ith layer is selected, VOPs of lower than the ith layer are also selected. Conversely, if a VOP of the ith layer is not selected, VOPs of higher than the ith layer cannot be selected, since we already run out of transmission resources.

Sometimes, a startup latency is needed to prefetch a few initial VOPs. Otherwise, it is likely to discard the beginning part of the video, unless the initial network bandwidth is large. The startup latency is accomplished by shifting the encoded video by σ, such that

$$x_i[k] = \begin{cases} 0, & 0 < k \le \sigma \\ x'_i[k-\sigma], & k > \sigma \end{cases},$$

where $x'_i[k-\sigma]$ is the VOP size of original encoded video.

3) Discard State: In the `discard` state, a sender needs to drop as many VOPs as possible to reduce the quality variability. However, as dropping many VOPs leads to the under-utilization of transmission resources, the sender has to return to the `select` state when a certain condition is satisfied. In Fig. 6, the sender leaves the `discard` state when the available cumulative capacity exceeds a threshold. The *available cumulative capacity* is defined by the difference of the cumulative capacity and the cumulative selected data (e.g., the available cumulative capacity exceeds a threshold d at k_1 in Fig. 5, and the sender leaves the `discard` state at this point). The question here is how to determine the threshold. Theorem 1 in Section IV-B.4 states that a threshold equal to the receiver buffer size achieves both consistent video quality and the necessary condition of the maximum network utilization.

Note that the selected data $s_i[k]$ is zero in the `discard` state, since no VOP is selected for transmission. On the contrary, the available cumulative capacity is not zero although it is smaller than the threshold. We can take advantage of the available capacity for prefetching next selected VOPs.

The optimal quality adaptation algorithm is shown in Fig. 7. This is an implementation of the state machine in Fig. 6 with the threshold value of the receiver buffer size. Note that this algorithm assumes the available network bandwidth information is known *a priori*. When there is a transition from the `select` state to the `discard` state, it is necessary to determine the next prefetch point. Line 16 implies that the point can be determined, only when the available network bandwidth information is available.

In optimal quality adaptation, lower layers will exhibit less quality fluctuation and better bandwidth utilization than higher layers, since quality adaptation is applied from the lowest layer to the highest layer. The performance of the algorithm depends on both the receiver buffer size b_i and the available network bandwidth $r_i[k]$, since the capacity of transmission resources is determined by the receiver buffer size and the available bandwidth. *UpdateBandwidth* computes the residual available bandwidth for higher layers.

4) Optimality of Algorithm: The optimality of quality adaptation is defined by minimum quality variability as long as the necessary condition of maximum network utilization holds. To achieve this goal, an optimal adaptation framework is proposed in Section IV-B.1. Based on this framework, we need to maximize the sojourn time in the `select` state and the `discard` state for minimum quality variability. Theorem 1 states that a threshold value equal to the receiver buffer size

```
1: procedure OptimalAdaptation (r[], x_i[], b_i)
2:      Initialization: r_1[k] = r[k], S_i[k] = 0, C_i[k] = 0
3:          where i ∈ {1, ..., L} and k ∈ {1, ..., N}
4:      for i = 1 to L do
5:          for k = 1 to N do
6:              C_i[k] = min(S_i[k-1] + b_i, C_i[k-1] + r_i[k])
7:              if s_i[k-1] = x_i[k-1] then
8:                  if C_i[k] ≥ S_i[k-1] + x_i[k] then
9:                      s_i[k] = x_i[k]
10:                     S_i[k] = S_i[k-1] + x_i[k]
11:                 else
12:                     s_i[k] = 0
13:                     S_i[k] = S_i[k-1]
14:                 endif
15:             else
16:                 if C_i[k] ≥ S_i[k-1] + b_i then
17:                     s_i[k] = x_i[k]
18:                     S_i[k] = S_i[k-1] + x_i[k]
19:                 else
20:                     s_i[k] = 0
21:                     S_i[k] = S_i[k-1]
22:                 endif
23:             endif
24:         enddo
25:         UpdateBandwidth(r[], C_i[], S_i[])
26:     enddo
27: endprocedure
```

Fig. 7. Optimal quality adaptation algorithm. This is an implementation of the state transition diagram with the threshold of receiver buffer size.

achieves maximally consistent video quality under the constraint of transmission resources utilization. Theorem 1 also implies that the quality adaptation algorithm should exploit transmission resources as much as possible to achieve both minimum quality variation and the necessary condition of maximum network utilization.

Theorem 1: In the framework of the optimal quality adaptation, a threshold value equal to the *receiver buffer size* satisfies 1) minimum video quality variability and 2) the necessary condition of maximum network utilization.

Proof: see Appendix A. ∎

C. Online Heuristic

The optimal quality adaptation algorithm in Section IV-B assumes the available bandwidth information is known in advance. Since the Internet does not provide any information about the bandwidth evolution, we need an algorithm that minimizes quality variability without using future bandwidth information. In this section, we develop an online heuristic to satisfy this requirement.

The online heuristic algorithm is shown in Fig. 8. The framework of online heuristic is similar to the optimal adaptation algorithm, since it is also developed based on the state machine in Fig. 6. The differences between the online heuristic and the optimal adaptation are 1) the online heuristic makes a decision on which layer and which VOP to be transmitted in real time (lines 4-6), and 2) a sender makes a receiver prefetch

the next selected VOPs when there is a transition from the `select` state to the `discard` state (line 15).

```
 1: procedure OnlineAdaptation (r[], x_i[], b_i)
 2:     Initialization: S_i[k] = 0, C_i[k] = 0
 3:         where i ∈ {1, ..., L} and k ∈ {1, ..., N}
 4:     for k = 1 to N do
 5:         r̂[k] = SmoothBandwidth(r[k])
 6:         for i = 1 to L do
 7:             C_i[k] = min(S_i[k−1] + b_i, C_i[k−1] + r[k])
 8:             if s_i[k−1] = x_i[k−1] then
 9:                 if C_i[k] ≥ S_i[k−1] + x_i[k] then
10:                     s_i[k] = x_i[k]
11:                     S_i[k] = S_i[k−1] + x_i[k]
12:                 else
13:                     s_i[k] = 0
14:                     S_i[k] = S_i[k−1]
15:                     NextSelect = EstimateNextSelect(r̂[k])
16:                 endif
17:             else
18:                 if k ≥ NextSelect then
19:                     s_i[k] = x_i[k]
20:                     S_i[k] = S_i[k−1] + x_i[k]
21:                 else
22:                     s_i[k] = 0
23:                     S_i[k] = S_i[k−1]
24:                 endif
25:             endif
26:             UpdateBandwidth(r[], C_i[], S_i[])
27:         enddo
28:     enddo
29: endprocedure
```

Fig. 8. Online heuristics. This algorithm performs quality adaptation without using the future bandwidth information.

The question is how to determine the next prefetch point at the transition time. We consider an MA (Moving Average) type estimator to determine the prefetch point. The MA estimator is simple and widely known for the usage of TCP retransmission timeout estimation in [7]. Using MA estimator, the available bandwidth can be estimated by the weighted sum of smoothed average and the mean deviation. The estimated available bandwidth, $\hat{r}$, is given by the following relationship and is implemented in the *SmoothBandwidth* function:

$$
\begin{aligned}
err &\leftarrow r[k] - sr \\
sr &\leftarrow sr + 0.125 \cdot err \\
d &\leftarrow d + 0.25 \cdot (|err| - d) \\
\hat{r} &\leftarrow sr + 4 \cdot d
\end{aligned}
$$

The next prefetched VOP is the one that occurs at the point determined by $\min([\frac{b_i}{\hat{r}}], M)$ in the *EstimateNextSelect* function, where M is the maximum duration for the next selected VOP. This relationship is a linear approximation of the point of the next prefetched VOP, when the threshold is equal to the receiver buffer size. We limit the duration to M to prevent small $\hat{r}$ from causing the under-utilization of transmission resources.

It should be noted that the performance of the online heuristic algorithm depends on the available bandwidth estimator.

Therefore, we need to ensure that a prediction based on the estimator is carried out effectively. The author in [15] investigated the characteristics of the Internet delay variation. The measurement results show that the variation ranges primarily on time scales of $0.1 - 1$ seconds, although frequently the time scales can be much larger. The authors in [21] investigated the stationarity of the Internet. One of the most interesting findings in the paper is that the stationarity of Internet path properties depends on time scales. Experimental results show that the stationarity of packet loss rate is well preserved on time scales of a few seconds to minutes. Since the throughput of a transport protocol is mainly determined by delay and loss rate [5], [14], we can expect that the throughput stationarity is maintained on time scales of a few seconds to minutes. The results in [21] show that we can expect a few minutes of stationarity to prevent available bandwidth from varying by more than $\pm 10\%$. Therefore, we can expect good performance of the bandwidth estimator by limiting the region of operating points (i.e., by setting the maximum duration M within a few seconds or a few minutes).

V. EVALUATION

In this section, we show results from experiments by which we evaluate our algorithms (the offline optimal algorithm and the online heuristic) and compare it to the baseline "composed" algorithm. The main goal is to evaluate the performance of the algorithms and the impact of parameters, such as the receiver buffer size or transport protocols, on the perceptual video quality. The experimental results including encoded video traces, network simulation script, and decoded video are available at the companion web site [3].

In the experiments, we use the *A river runs through it* stream which is generated in Section III. MPEG-4 FGS defines a requirement that a streaming video sender has the flexibility to send any portion of the enhancement layer to provide fine granularity of control [16]. However, how to manipulate the truncated bit stream at FGS decoder is not standardized [9]. To compromise between the requirement and the implementation, we assume that an FGS codec does not decode an incomplete VOP, hence the granularity of control is given by a VOP size.

a) Bandwidth Variability: Fig. 9 (a) shows the network topology in the experiment which has a single bottleneck link whose capacity is 10Mbps and link delay is 50ms. We consider the use of our algorithms on over both TCP and TFRC/UDP (TCP-Friendly Rate Control) as shown in Figs. 9 (b) and (c) respectively. The following accounts for why the two transport protocols are employed for the experiment: TCP is the dominant protocol in the Internet. Although the stability of the Internet is achieved by end-to-end control of TCP, the nature of AIMD (Additive Increase Multiplicative Decrease) is regarded as inappropriate for streaming applications since it incurs significant data rate variability. Also, many video streaming applications simply transmit video data at the rate at which it was encoded, which may lead to congestion collapse of the Internet. TFRC was proposed in [5] to provide congestion control, fairness with TCP, and smoothly-changing
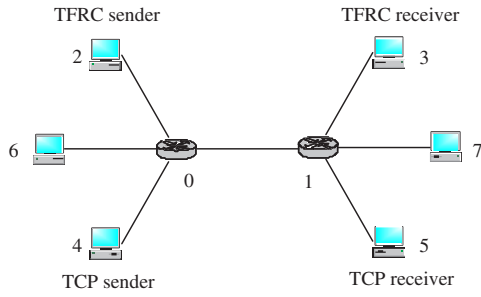
(a) Network topology

(b) Quality adaptation over TFRC

(c) Quality adaptation over TCP

Fig. 9. Experiment model. Network topology in (a) has a single bottleneck link. Quality adaptations are performed over TFRC and TCP in (b) and (c).

data rate for streaming applications. Therefore, we investigate the performance over TCP, the main player of the Internet, and TFRC, an improved transport protocol for streaming media.

We generate the bandwidth variability resulting from each underlying transport protocol using the network model shown in Fig. 9 (a). We run an `ns-2` [12] simulation of the network model (described below). To model bandwidth variability in our experiments, we use the throughput experienced between the TFRC sender and receiver (nodes 2 and 3) or the throughput experienced between the TCP sender and receiver (nodes 4 and 5). TFRC throughput is measured by counting the number of packets from node 1 to node 3 (see Fig. 10), and TCP throughput is measured between node 1 and node 5 (see Fig. 15). To simulate real background traffic, a flow is generated between nodes 6 and 7 by superposing 100 ON/OFF sources of pseudo nodes which have Pareto distribution [20].

To accommodate received packets, we need to allocate buffer space to each layer. We allocate buffer size in the ratio of $1 : 8 : 16$ for base, FGS and FGST layers, respectively, i.e., 4% of whole buffer space is assigned to the base layer, 32% to FGS layer, and 64% to FGST layer[1].

A. Performance over TFRC

The bandwidth variability of TFRC is shown in Fig. 10. As described in [1], [5], TFRC is slowly responsive to packet loss, which results in reduced variability of available bandwidth. However, the sluggishness of TFRC leads to an unnecessary decrease in throughput and an increase in response time under dynamic environment [1]. Fig. 10 demonstrates that steady state throughput is about 4Mbps, slightly lower than the fair share, and that response time to reach the steady state is greater than 20 seconds.

[1]Note that this provides more buffering for base layer than would be proposed in [19]. We found this to be necessary to protect against losses resulting from burstiness of the base layer.
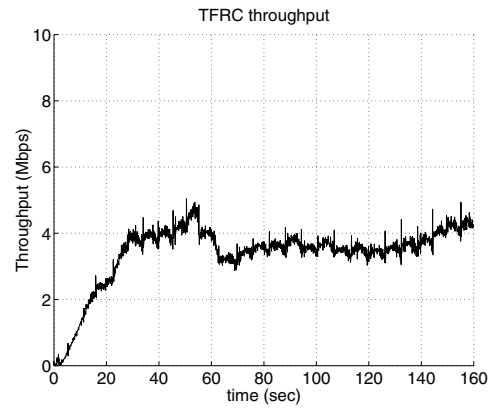

Fig. 10. TFRC throughput. TFRC exhibits small rate variability, slow response time, slightly less throughput compared with TCP.

In Fig. 11 and 12, we present experiment results of the composed algorithm and the optimal adaptation algorithm which assume the knowledge of available bandwidth information. In the composed algorithm, we allocate half of the receiver buffer to the rate smoothing buffer and the remaining half to the quality smoothing buffer. Fig. 11 shows that the composed algorithm can reduce the quality variability by increasing the receiver buffer size. The quality transition in the second enhancement layer is 121 for 600kB buffer, but it is reduced to 13 for 6MB buffer, where the *quality transition* of the ith layer is defined by $\sum_{k=1}^{N} I_i(k)$, where

$$I_i(k) = \begin{cases} 1, & \text{if } s_i[k-1] \neq s_i[k] \\ 0, & \text{otherwise.} \end{cases}$$
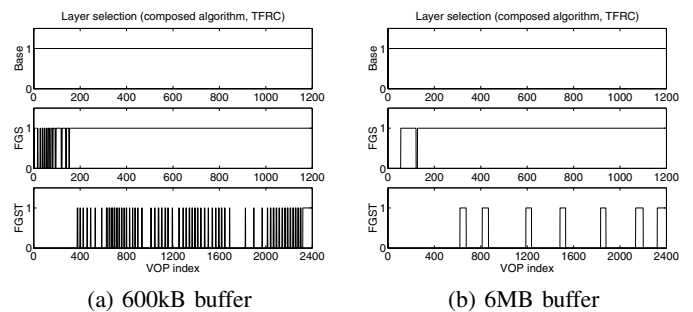

(a) 600kB buffer

(b) 6MB buffer

Fig. 11. Performance of the composed algorithm over TFRC

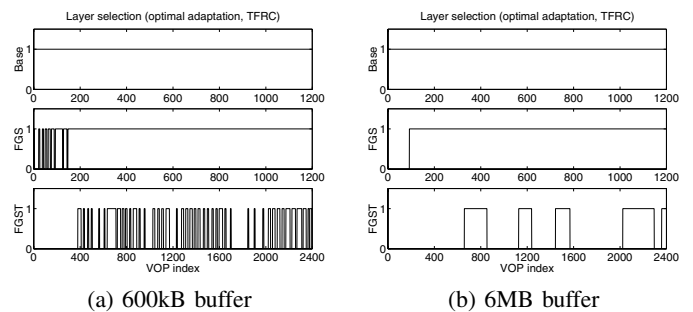(a) 600kB buffer

(b) 6MB buffer

Fig. 12. Performance of the optimal adaptation algorithm over TFRC

Fig. 12 shows the results of the optimal quality adaptation algorithm. Compared with Fig. 11, the results show that the quality variability is significantly decreased and the length of run is increased, where a *run* is defined by a sequence of consecutive VOPs. The quality transition can also be decreased when we increase the receiver buffer size: There are 87 transitions in the FGST layer for 600kB buffer, but transitions are reduced to 9 for 6MB buffer. Note that a lot of initial VOPs in enhancement layers are dropped because of the slow response time of TFRC. Although the initial latency is set to 5 seconds, it is not enough to accommodate enhancement layers.

We also compare the threshold-based streaming algorithm [19] with the online heuristic (Note that both do not require knowledge of future bandwidth). The authors in [19] proposed a threshold policy for finite length video streaming that tries to minimize loss probability in the base and the enhancement layer. Although the original policy considered only two layers, we extend it to three layers for the purposes of our comparison. The details of the extension are shown in Appendix B.
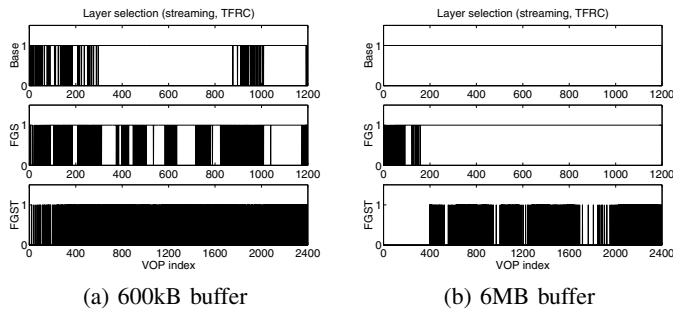

(a) 600kB buffer (b) 6MB buffer

Fig. 13. Performance of threshold-based streaming over TFRC
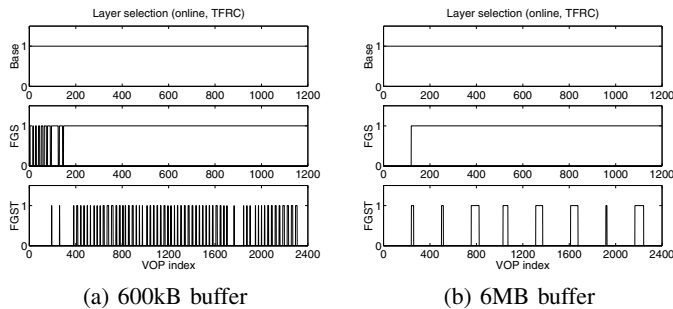

(a) 600kB buffer (b) 6MB buffer

Fig. 14. Performance of the online heuristic over TFRC

Fig. 13 shows the performance of the threshold-based streaming. We choose $\alpha_b = \alpha_{FGS} = 0.2$ and $q_b = 0.2 \cdot b_1$, $q_{FGS} = 0.2 \cdot b_2$, where b_1 is the buffer size of the base layer and b_2 is the buffer size of the FGS layer. Fig. 13 exhibits a lot of quality transitions and a small average run. Even if we increase the buffer size by 6MB, a lot of quality transitions still remain in the FGST layer. The reason is that the objective of the threshold policy is to minimize loss probability, not to maximize perceptual quality.

The performance of the online heuristic in Fig. 14 exhibits 126 quality transitions in the FGST layer for 600kB buffer, however it is reduced to 16 for 6MB buffer. Compared with

Fig. 11 and 12, we can find that the performance of online adaptation is not so good as optimal adaptation, but it is comparable with the composed algorithm.
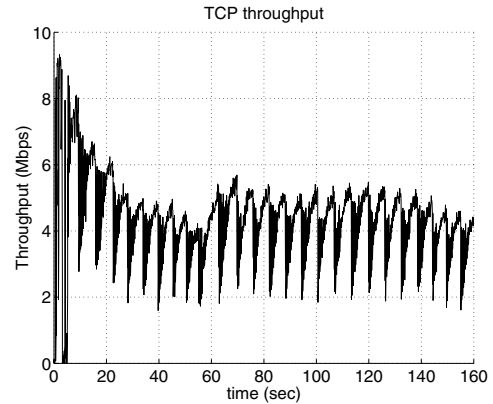
B. Performance over TCP

Fig. 15. TCP throughput. TCP exhibits fast response time, more throughput, and large rate variability.

Fig. 15 shows TCP throughput from the network simulation. We use this throughput as a model of bandwidth variability. As reported in [1], a TCP flow achieves more throughput than a TFRC flow in dynamic network conditions. Fast responsiveness of TCP leads to an aggressive behavior before it reaches the steady state. Even in the steady state, TCP achieves slightly more throughput than TFRC. However, small time scale variability is significant as much as about 3Mbps in the steady state, since TCP is very sensitive to packet losses.
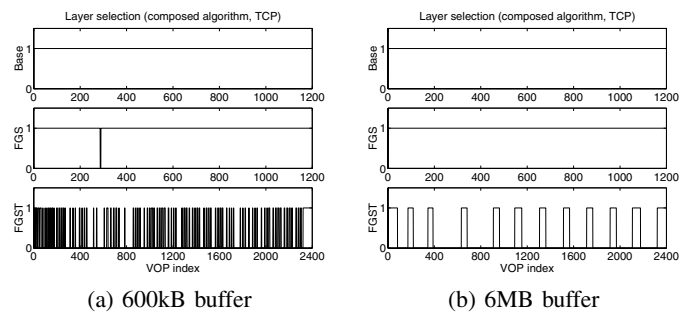

(a) 600kB buffer (b) 6MB buffer

Fig. 16. Performance of the composed algorithm over TCP

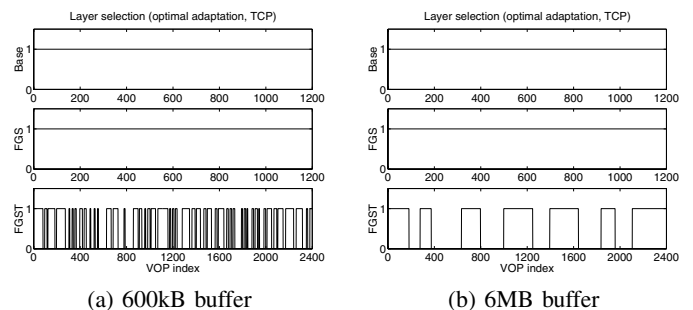(a) 600kB buffer (b) 6MB buffer

Fig. 17. Performance of the optimal adaptation algorithm over TCP

Figs. 16 and 17 show the performance of the composed algorithm and the optimal adaptation algorithm when running over TCP. Compared with Fig. 11 and 12, we can find that the performance is significantly improved, especially in the transient state. The composed algorithm exhibits only two quality transitions in the FGS layer when the receiver buffer size is 600kB. In the other cases, there are no quality transitions in the FGS layer. We can also find that quality transitions are reduced and the length of run is increased, as the receiver buffer size increases. Note that the performance in the steady state is still better than TFRC. For example, if we consider the FGST layer after 400 VOPs in optimal adaptation, the number of quality transitions in Fig. 12 (b) is equal to Fig. 17 (b) but the average run length of TCP is longer than TFRC.

Two reasons contribute to the superiority of TCP: 1) TCP achieves more throughput than TFRC in dynamic condition, 2) although TCP exhibits significant small time scale variability, it can be successfully accommodated by the receiver buffer.

Fig. 18 and 19 show the performance of the threshold-based streaming and the online heuristic which do not require prior knowledge of the available bandwidth information. The results show that the threshold-based streaming exhibits a lot of quality transitions when the receiver buffer size is small. Even if we increase the buffer size by 6MB, there are still many quality transitions in the FGST layer. In Fig. 19, the online heuristic does not show any quality transitions in both the base and the FGS layers. Compared with Fig. 16, we can find that the online heuristic shows similar performance to the composed algorithm.
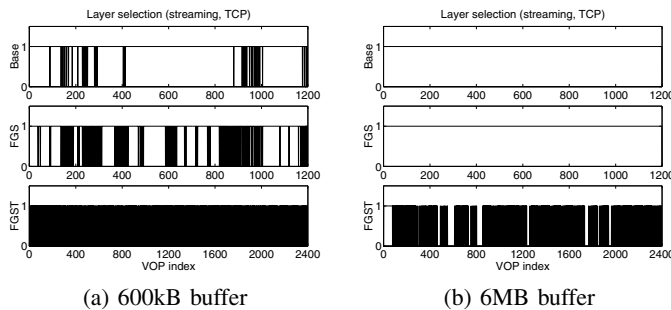

(a) 600kB buffer (b) 6MB buffer

Fig. 18. Performance of threshold-based streaming over TCP

(a) 600kB buffer (b) 6MB buffer
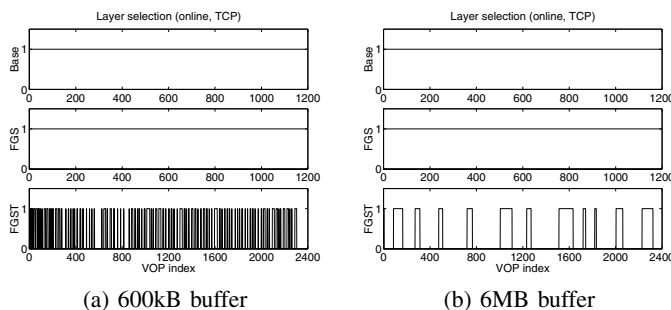
Fig. 19. Performance of the online heuristic over TCP

We also investigate the performance for other video se-

quences: *A river runs through it, Jurassic park I, Starwars: episode I*, and *Red corner*. We measure the performance in terms of the AQT (Average Quality Transition) and the ARL (Average Run Length) [13]:

$$
\begin{aligned}
AQT &= \frac{1}{L}\sum_{i=1}^{L}\sum_{k=1}^{N} I_i(k) \\
ARL &= \frac{1}{L}\sum_{i=1}^{L}\frac{\sum_{j=1}^{k_i} n_j}{k_i},
\end{aligned}
$$

where $I_i(k)$ is defined in Section V-A, k_i is the number of runs in the ith layer, and n_j is the length of the jth run.

Table I shows the performance of all algorithms. In all cases, the optimal adaptation algorithm exhibits the smallest AQT, since the algorithm is optimized to minimize quality variability. Moreover, the optimal adaptation algorithm shows the largest ARL. Hence, we can expect the optimal adaptation generates the longest video with the smallest quality transitions. Note that AQT of TFRC is greater than TCP in most cases and ARL of TFRC is smaller than TCP, since TCP possesses faster response time and higher throughput than TFRC. The results also show that the performance of the online heuristic is not as good as optimal adaptation, but it is nearly equal to that of the composed algorithm. This is not surprising since both the optimal and composed algorithms operate with a knowledge of future bandwidth variability.

VI. Conclusion

In this paper, we consider the problem of providing perceptually good quality for layered VBR streaming video. The problem is challenging because both the transmission resources and the encoded video exhibit multiple time scale variability. To accommodate the variability, we develop an optimal adaptation algorithm that minimizes quality variability while increasing the usage of the available bandwidth. We then propose an online heuristic which does not require knowledge of future bandwidth variability. Experimental results demonstrate that quality adaptation algorithms reduce the quality variability significantly and that the optimal quality adaptation algorithm exhibits the best performance. We also show that our quality adaptation provides better results when running over TCP as compared with TFRC/UDP. This is because TCP achieves higher throughput and the receiver buffer can accommodate the short-term variability of TCP. Because perceptual quality is hard to assess by graphs and numbers alone, we have provided videos on a companion web site [3] which demonstrates the perceptual differences among the different algorithms and options.

References

[1] D. Bansal, H. Balakrishnan, S. Floyd, and S. Shenker, "Dynamic behavior of slowly responsive congestion control algorithms," *Proceedings of ACM SIGCOMM 2001*, San Diego, CA, Aug. 2001.

[2] *Coding of Audio-Visual Objects, Part-2 Visual, Amendment 4: Streaming Video Profile*, ISO/IEC 14496-2/FPDAM4, July 2000.

TABLE I
EXPERIMENT RESULTS FOR 4 VIDEO STREAMS

	Buffer size		600kB				6MB			
	Algorithm		streaming	online	composed	optimal	streaming	online	composed	optimal
AQT	*A river runs*	TFRC	642	48	49	**34**	282	5	5	**3**
	through it	TCP	666	64	62	**31**	396	7	8	**4**
	Jurassic park	TFRC	500	45	57	**35**	404	5	6	**3**
		TCP	537	65	63	**25**	469	7	8	**2**
	Starwars:	TFRC	511	45	60	**31**	490	5	6	**2**
	episode I	TCP	514	57	57	**16**	456	6	7	**1**
	Red corner	TFRC	533	44	53	**31**	374	5	5	**3**
		TCP	575	60	60	**21**	454	7	8	**2**
ARL	*A river runs*	TFRC	13	877	858	**890**	821	1534	1197	**1587**
	through it	TCP	22	1601	1199	**1608**	1599	1617	1617	**1663**
	Jurassic park	TFRC	807	931	897	**936**	829	1546	1596	**1619**
		TCP	820	1602	1601	**1614**	1599	1625	1626	**1722**
	Starwars:	TFRC	823	959	957	**1006**	835	1554	1605	**1667**
	episode I	TCP	835	1603	1602	**1624**	1599	1636	1635	**1962**
	Red corner	TFRC	47	958	930	**1002**	833	1547	1602	**1616**
		TCP	82	1602	1601	**1616**	1599	1628	1625	**1720**

[3] Companion web site, `http://www.cc.gatech.edu/computing/Telecom/people/Phd/tkim/qa.html`, 2002.

[4] P. de Cuetos and K. W. Ross, "Adaptive rate control for streaming stored fine grained scalable video," *Proceedings of NOSSDAV 2002*, Miami, FL, May 2002.

[5] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation based congestion control for unicast applications," *Proceedings of ACM SIGCOMM 2000*, Stockholm, Sweden, Aug. 2000.

[6] M. Grossglauser, S. Keshav, and D. N. C. Tse, "RCBR: A simple and efficient service for multiple time-scale traffic," *IEEE/ACM Trans. Networking*, vol. 5, pp. 741-755, Dec. 1997.

[7] V. Jacobson, "Congestion avoidance and control," *Proceedings of ACM SIGCOMM '88*, Stanford, CA, Aug. 1988.

[8] T. Kim and M. H. Ammar, "A comparison of layering and stream replication video multicast schemes," *Proceedings of NOSSDAV 2001*, Port Jefferson, NY, June 2001.

[9] W. Li, "Overview of fine granularity scalability in MPEG-4 video standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 3, pp. 301-317, Mar. 2001.

[10] X. Li, S. Paul, and M. H. Ammar, "Layered video multicast with retransmission (LVMR): Evaluation of hierarchical rate control," *Proceedings of IEEE INFOCOM '98*, San Francisco, CA, Mar. 1998.

[11] S. McCanne, V. Jacobson, and M. Vetterli, "Receiver driven layered multicast," *Proceedings of ACM SIGCOMM '96*, Stanford, CA, Aug. 1996.

[12] ns-2 network simulator, `http://www.isi.edu/nsnam/ns/`, 2001.

[13] S. Nelakuditi, R. R. Harinath, E. Kusmierek, and Z.-L. Zhang, "Providing smoother quality layered video stream," *Proceedings of NOSSDAV 2000*, Chapel Hill, NC, June 2000.

[14] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP reno performance: A simple model and its empirical validation," *IEEE/ACM Trans. Networking*, vol. 8, no. 2, pp. 133-145, Apr. 2000.

[15] V. Paxson, "End-to-end Internet packet dynamics," *IEEE/ACM Trans. Networking*, vol. 7, no. 3, pp. 277-292, June 1999.

[16] H. Radha, M. van der Schaar, and Y. Chen, "The MPEG-4 fine-grained scalable video coding method for multimedia streaming over IP," *IEEE Trans. Multimedia*, vol. 3, no. 1, Mar. 2001.

[17] R. Rejaie, D. Estrin, and M. Handley, "Quality adaptation for congestion controlled video playback over the Internet," *Proceedings of ACM SIGCOMM '99*, Cambridge, MA, Aug. 1999.

[18] J. D. Salehi, Z.-L. Zhang, J. Kurose, and D. Towsley, "Supporting stored video: Reducing rate variability and end-to-end resource requirements through optimal smoothing," *IEEE/ACM Trans. Networking*, vol. 6, no. 4, Aug. 1998.

[19] D. Saparilla and K. W. Ross, "Optimal streaming of video," *Proceedings of INFOCOM 2000*, Tel Aviv, Israel, Mar. 2000.

[20] W. Willinger, M. S. Taqqu, R. Sherman, and D. V. Wilson, "Self-similarity through high-variability: Statistical analysis of Ethernet LAN traffic at the source level," *Proceedings of ACM SIGCOMM '95*, Cambridge, MA, Aug. 1995.

[21] Y. Zhang, V. Paxson, and S. Shenker, "The stationarity of internet path properties: Routing, loss, and throughput," *ACIRI technical report*, May 2000.

APPENDIX A   PROOF OF THEOREM 1

We employ a continuous-time model to prove Theorem 1. Our notation for the continuous version of the variables is as follows:

$x_i(t)$	VOP size of the ith layer at time t
$X_i(t)$	cumulative data requirement: $X_i(t) = \int_0^t x_i(s)ds$
$s_i(t)$	selected data of the ith layer at time t:
	$s_i(t) = \begin{cases} x_i(t), & \text{for the } \texttt{select} \text{ state} \\ 0, & \text{otherwise} \end{cases}$
$S_i(t)$	cumulative selected data: $S_i(t) = \int_0^t s_i(u)du$
b_i	receiver buffer size to store unplayed ith layer video
$r_i(t)$	available bandwidth
$R_i(t)$	cumulative bandwidth constraint: $R_i(t) = \int_0^t r_i(s)ds$
$C_i(t)$	cumulative capacity of the ith layer:
	$C_i(t) = \min\{S_i(t) + b_i, R_i(t)\}$
$T_i(t)$	size of transmitted data
$U_i(t)$	network bandwidth utilization: $U_i(t) = \frac{T_i(t)}{R_i(t)}$

We first present a condition that achieves minimum quality variability. It is evident that the variability increases, as the number of transitions between the `select` state and the `discard` state increases during a fixed interval. Hence, long average sojourn time reduces average number of state transitions, which leads to less variability. Our task is to determine

the threshold value of the quality adaptation framework to maximize average sojourn time.

Assume that we begin with enough network bandwidth, but we run out of capacity at t_0, which results in $X_i(t_0) = C_i(t_0)$ as shown in Fig. 20. Let u be the sojourn time in the discard state. Then, a sender remains in the discard state until t_0+u, such that $s_i(\eta) = 0$ for $t_0 \leq \eta < t_0 + u$. At time $t_0 + u$, the sender returns to the select state, since enough capacity is accumulated. Given a threshold d,

$$C_i(t_0 + u) - S_i(t_0 + u) = d \tag{1}$$

cumulative data in the ith layer

$X_i(t)$

d

d'

$C_i(t)$

$S_i(t)$

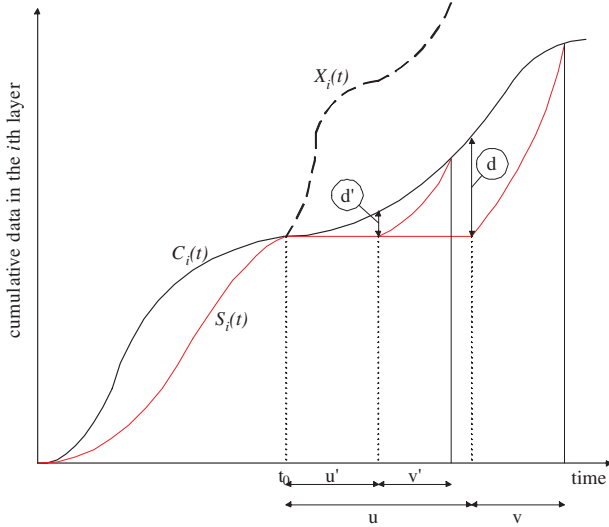t_0 u' v' time

u v

Fig. 20.   Illustration of quality adaptation

Let v be the sojourn time in the select state. With the threshold value d, we need to find the average sojourn time in the select state $E\{v\}$. The sender is assumed to enter the select state at t_0+u and leave the state at t_0+u+v, which results in running out of capacity at $t_0 + u + v$. Hence, $S_i(t_0+u+v) = C_i(t_0 + u + v)$. Using (1),

$\int_{t_0+u}^{t_0+u+v} x_i(\eta)d\eta = \int_{t_0+u}^{t_0+u+v} s_i(\eta)d\eta$
$= \int_{t_0+u}^{t_0+u+v} s_i(\eta)d\eta + C_i(t_0 + u + v) - S_i(t_0 + u + v)$
$= C_i(t_0 + u) - S_i(t_0 + u) + \int_{t_0+u}^{t_0+u+v} c_i(\eta)d\eta$
$= d + \int_{t_0+u}^{t_0+u+v} c_i(\eta)d\eta.$

If we assume $x_i(t)$, $c_i(t)$, and v are independent, conditional average yields $E\{v\}E\{x_i(t)\} = d + E\{v\}E\{c_i(t)\}$. Hence,

$$E\{v\} = \frac{d}{E\{x_i(t)\} - E\{c_i(t)\}}. \tag{2}$$

Consider a different quality adaptation scheme with threshold d', such that $d' < d$. Let u' be the sojourn time in the discard state, and v' be the sojourn time in the select state. From (1) and (2), we can get

$C_i(t_0 + u') = S_i(t_0 + u') + d'$
$E\{v'\} = \frac{d'}{E\{x_i(t)\} - E\{c_i(t)\}}$

Since $S_i(t_0 + u) = S_i(t_0 + u')$ and $C_i(t)$ is non-decreasing,

$$\begin{aligned} u &\geq u' \\ E\{v\} &> E\{v'\}. \end{aligned} \tag{3}$$

From (3), as we increase the threshold value, average sojourn time spent in each state gets longer and we can achieve less variability. Therefore, the optimal threshold that achieves minimum quality variability is given by

$$\tau = \max_{\tau_i}\{\tau_i \mid \tau_i \in T, T \text{ is a set of feasible thresholds}\}. \tag{4}$$

Now we consider a threshold value that maximizes the network bandwidth utilization. Since the maximum sending rate is bounded by both the network bandwidth and the receiver buffer size, $T_i(t) \leq C_i(t) = \min\{S_i(t) + b_i, R_i(t)\}$. Therefore, utilization at time t is bounded by

$$\begin{aligned} U_i(t) &= \frac{T_i(t)}{R_i(t)} \\ &\leq \frac{C_i(t)}{R_i(t)} \\ &= \frac{\min\{S_i(t)+b_i, R_i(t)\}}{R_i(t)} \\ &= \begin{cases} \frac{S_i(t)+b_i}{R_i(t)}, & \text{if } S_i(t) + b_i < R_i(t) \\ 1, & \text{otherwise.} \end{cases} \end{aligned}$$

The necessary condition of $U_i(t) = 1$ is given by $S_i(t) + b_i \geq R_i(t)$ for all t. Since $C_i(t) = \min\{S_i(t) + b_i, R_i(t)\}$,

$$\begin{aligned} b_i &\geq R_i(t) - S_i(t) \\ &\geq C_i(t) - S_i(t). \end{aligned} \tag{5}$$

From (1), (5) implies

$$b_i \geq \tau, \tag{6}$$

which presents the threshold value τ should be equal or less than the receiver buffer size to achieve maximum bandwidth utilization.

From (4) and (6), the optimal threshold value is given by the receiver buffer size,

$$\tau = b_i.$$

APPENDIX B   THREE-LAYER EXTENSION OF THRESHOLD-BASED STREAMING

$$\begin{cases} \pi_b[k] = 1, \pi_{FGS}[k] = 0, & \text{if } Y_b[k] < q_b, \\ \pi_b[k] = \alpha_b, \pi_{FGS}[k] = 1 - \alpha_b, \\ \qquad \text{if } Y_b[k] \geq q_b, Y_{FGS}[k] < q_{FGS}, \\ \pi_b[k] = \alpha_b, \pi_{FGS}[k] = \alpha_{FGS}, \\ \qquad \text{if } Y_b[k] \geq q_b, Y_{FGS}[k] \geq q_{FGS}, \\ \pi_b[k] = \alpha_b, \pi_{FGS}[k] = 0, \\ \qquad \text{if } Y_b[k] \geq q_b, Y_{FGS}[k] > \sum_{j=k}^{N} x_{FGS}[j], \\ \pi_b[k] = 0, \pi_{FGS} = 1, \\ \qquad \text{if } Y_b[k] > \sum_{j=k}^{N} x_b[j], Y_{FGS}[k] < q_{FGS}, \\ \pi_b[k] = 0, \pi_{FGS}[k] = \alpha_{FGS}, \\ \qquad \text{if } Y_b[k] > \sum_{j=k}^{N} x_b[j], Y_{FGS}[k] \geq q_{FGS}, \\ \pi_b[k] = 0, \pi_{FGS}[k] = 0, \\ \text{if } Y_b[k] > \sum_{j=k}^{N} x_b[j], Y_{FGS}[k] > \sum_{j=k}^{N} x_{FGS}[j], \end{cases}$$

where $\pi_b[k]$, $\pi_{FGS}[k]$, and $\pi_{FGST}[k]$ are fraction of transmission resources; $x_b[k]$ and $x_{FGS}[k]$ are encoded data rate; $Y_b[k]$ and $Y_{FGS}[k]$ are buffer occupancy. Note that $\pi_b[k] + \pi_{FGS}[k] + \pi_{FGST}[k] = 1$.