Balancing Interruption Frequency and Buffering Penalties in VBR Video Streaming

Guanfeng Liang and Ben Liang
Dept. Electrical and Computer Engineering
University of Toronto
Email: {guanfeng, liang}@comm.utoronto.ca

***Abstract*—The main goal of a streaming application is to enable the successful decoding of each video object before its displaying deadline is violated, and to recover from a deadline violation properly. Hence, we define the main performance metric of a streaming system as the number of interruptions during a video presentation, or the number of jitters. Previous literature has described solutions to estimate the jitter-free probability for an entire video segment. In this work, we present a novel analytical framework, which requires only a Markov Variable Bit Rate (VBR) channel model, to study the *frequency* of jitters under the constraint of initial playback delay, receiver buffer size, and different jitter recovering schemes. Both the infinite and finite buffer cases are considered. This technique is then applied to investigate streaming over a wireless system modeled by an extended Gilbert channel with ARQ transmission control. Experimental results with MPEG-4 VBR encoded video validate our analysis. Finally, we show that the proposed analysis provides a theoretical foundation to quantify the tradeoffs between the jitter frequency, jitter recovering delay, initial delay, and the receiver buffer size for a general class of VBR streaming over random VBR channels with different jitter recovering schemes.**

I. Introduction

Mobile devices of the near future are expected to bring ubiquitous access to streaming multimedia services, such as TV news, music video, and online movie. Streaming multimedia are likely to become major applications in future mobile systems and may indeed be a key factor for their success. However, wireless multimedia delivery faces several challenges, such as bandwidth scarcity, random channel variation, and limited storage capacity [1], [2]. In this work, we focus on the streaming of pre-encoded video to wireless clients, taking into account these system constraints and limited resources.

A general media streaming system is illustrated in Figure 1. This system consists of a media streaming *server*, a *transport channel*, and a streaming *client*. The server stores a number of pre-encoded video objects in the wired backbone network. We consider general variable bit-rate (VBR) encoded videos, with non-linear playback curves [3]. These video objects are accessed by the clients, through a wire or wireless access network.

In general, the transmission rate of the channel varies over time. Video display interruption may occur if data are not delivered on time when the transmission rate does not match the encoded rate. We term the event of playback interruption *playout jitter*, or *jitter*. Clearly, jitter reduces the perceived video quality and is undesirable in video streaming.

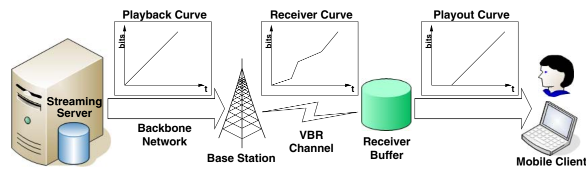


Fig. 1. A typical wireless media streaming system.

Depending on the channel condition and the chosen jitter recover scheme, a client may experience a number of jitters and some jitter buffering delay every time jitter occurs. From a client's point of view, the frequency of jitters experienced should not be too high, and the jitter-recovery delays should not be too long, either. Unfortunately, these two objectives conflict with each other. To address the balancing of this tradeoff is uniquely important to streaming media systems.

In this work, we present an analytical framework to quantify the fundamental tradeoffs between the frequency of jitters, jitter-recovery delay, the initial playout delay, and the receiver buffer size, in order to optimize the streaming of VBR encoded video over a random VBR channel. Our main contributions include the following: *First*, we propose an analytical framework to derive the distribution of the number of jitters, for three different jitterrecovery buffering schemes, and for both infinite and finite buffering cases. The proposed analysis technique provides a means to compute the frequency of jitters for a general class of VBR multimedia streaming over random VBR channels. *Second*, we apply the proposed analysis toward optimal streaming over a general wireless network. Numerical analysis results are obtained for wireless systems modeled by a generic Markov channel with ARQ transmission control. *Finally*, experiments using sample MPEG-4 video traces are carried out to validate the proposed analysis and provide new insight into the optimal balancing of delay, buffering, and jitters for optimal multimedia streaming.

The rest of this paper is organized as follows. We discuss related work in Section II. The network model is presented in Section III. In Section IV, an analytical framework is presented to derive the jitter frequency given different initial delay values and receiver buffer sizes. This framework is applied to study the performance of three different jitter-recovery buffering schemes. Section V provides the numerical and experimental results for streaming MPEG-4 videos over a wireless system and discusses the performance tradeoffs. Conclusions are drawn in Section VI.

II. Related Work

One obvious way to avoid jitter is to start displaying the video only after it is fully downloaded. By doing this, jitter is completely avoidable, but it also results in the longest initial delay and the maximal buffer size, which is generally unacceptable, especially for small wireless devices. In practice, the system buffers a certain portion of video data at the client before displaying the video, so that transient packet losses and delay do not constantly interrupt the playout of the stream. Intuitively, the more data is buffered, the fewer jitters will occur in the future, but the initial/jitter delay induced by buffering increases, too. For this reason, system designers must trade the reliability of uninterrupted playback against delay and buffer size.

In this context, Sen *et al.* proposed an online smoothing technique for VBR streaming video in [4] by introducing a few seconds of startup delay and a client buffer to compensate for the variation of video encoding rate. The authors of [5] used network calculus analysis to derive an optimal multimedia smooth scheme. However, both schemes require a dedicated smoothing server or an intermediate smoothing node and only consider a wired network offering guaranteed bandwidth service. Therefore these schemes are not suitable for error-prone networks such as wireless streaming systems.

The authors of [6] introduced the concept of having two buffers at the receiver - a delay jitter buffer and a decoder buffer. The delay jitter buffer is used to compensate for the delay jitter introduced by the channel and to reduce bit rate variations caused by the VBR behavior of the channel. By choosing a suitable initial delay, the jittered streaming data is de-jittered by the delay jitter buffer and a virtual constant bit rate (CBR) channel is formed at the input of the decoder buffer. Therefore, traditional hypothetical reference decoders (HRD) such as the video buffer verifier (VBV) for MPEG-2 or the H.263 HRD can be applied.

In [7], the authors compared the single receiver buffer approach with the aforementioned separate buffer approach and showed that a single receiver buffer always performs at least as good as two separate buffers. They then described a method to provide a certain QoS guarantee, where the initial delay and receiver buffer size are decided according to the upper and lower bounds of the random receiver curve to guarantee a minimum jitter-free probability. However, they did not give a general means to find such bounds of the receiver curve, and only a simple Bernoulli channel is considered.

A Markov chain analysis method was introduced in [8] to examine the tradeoff between buffer underflow probability and latency for Adaptive Media Playout (AMP) video streaming. Applying the two-state Gilbert-Elliott lossy channel model [9], [10], this method represents the streaming system with a Markov chain and gives the underflow probability by solving this Markov chain. However, in order to construct a Markov chain, the transmission time of each frame are assumed to be independent and identically distributed (i.i.d.) random variables, which is usually not true for VBR encoded video streaming where the frames have different sizes.

The problem of media streaming via TCP-Friendly Rate Control (TFRC) was studied in [11]. Modeling the TFRC traffic by a Markov-Renewal-Modulated Deterministic Process, the authors developed a queueing model for the TFRC client buffer, based on the imbedded Markov process of the buffer state immediately after a jitter. This model is then applied to obtain the distribution of the total duration of all rebuffering events experienced by a user. However, this work considers only CBR encoded videos and an infinite receiver buffer.

The authors of [12] considered the problem of supporting VCR functionaly in video on demand (VoD) systems. The reception process was modeled as a semi-Markov accumulation process, and a lower bound was obtained on the probability of successfully playout of the video with VCR actions. However, only CBR encoded videos were considered and the bound was not accurate for small value of segment length.

In [13], we proposed an analytical framework to bound the jitter probability for a video segment, given initial delay and receiver buffer size for VBR video streaming over VBR channels. It provides close bounds to the probability of jitter while only requiring knowledge of the VBR video playback curve, the maximum channel transmission rate, and general statistics of the channel. However, from the client's point of view, jitter frequency maybe a more meaningful QoS metric for longer video streams.

In this work, we propose a novel analytical framework to find the distribution of the number of jitters during the streaming of a VBR video over random VBR channels, given the jitter recovery scheme. The proposed framework can be applied to most general systems and only requires a Markovian model of the channel. To the best of our knowledge, this paper represents the first attempt to analytically quantify the the tradeoff between initial playback delay, receiver buffer size, jitter recovering operation, and the frequency of jitters for streaming with arbitrary encoding scheme and random channels.

III. Network Model

We consider the same video streaming system as in [7]. It consists of a video streaming server, a VBR transport channel, and a streaming client. Pre-encoded video objects are stored in the server. Each video object is characterized by a *playback curve* $p(t)$. The playback curve describes the total amount of data that have to be received by time t. It is generally assumed that $p(t) = 0$ for $t \leq 0$ and $p(t) = p(L)$ for $t \geq L$, where L is the length of the video in time and $p(L)$ is the size of the video in bits. The playback curve is assumed to be included in the preamble of the video stream and is available to the receiver.

Consider when a client requests a video object from the server. Corresponding to the request, the server streams video data to the client through the transport channel. The channel is assumed to be error free, possibly due to an ideal error control mechanism, such as coding or ARQ, but its bit rate may vary over time. The fluctuations in the transmission rate can lead

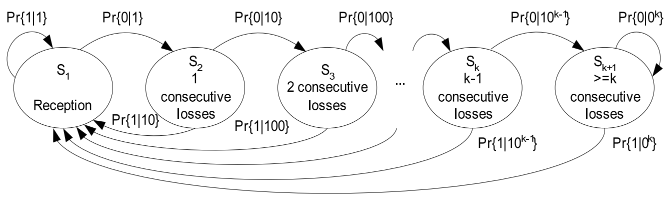


Fig. 2. A k-th order extended Gilbert model.

to significant late packet arrivals. The client allows an initial delay, which is a common practice in commercial streaming products. All packets arriving earlier than their playout times are stored in the client's local buffer.

In this work, we consider a generic class of discrete-time Markov channel models characterized by $(S, \mathbf{A}, R)$, where $S = \{S_1, ..., S_K\}$ is the set of channel states, $\mathbf{A}$ is the transition probability matrix, and $R = \{r_1, ..., r_K\}$ is the set of transmission rates associated the states. More precisely, r_i represents the number of packets that can be transmitted in one time slot.

Figure 2 shows an example of this class of Markov channel models. This is a k-th orded extended Gilbert model proposed by Sanneck *et al.* in [14]. It is a generalization of the Gilbert-Elliott model [9], [10], one of the most widely adopted wireless channel representations in the literature. There are $k + 1$ states, $\{S_1, S_2, ..., S_{k+1}\}$, where S_1 represents the reception state in which one packet is transmitted in one time slot, and $S_2, S_3, \ldots, S_{k+1}$ represent loss states, where no packet is transmitted successfully. For each state, the subscript represents the current distance from the last reception, except for state S_{k+1}, which represents the case that the current loss run-length is at least k, in which case the channel remains in state S_{k+1} with a subsequent loss or returns to S_1 with the first occurrence of reception. The extended Gilbert model out performs the traditional Gilbert-Elliott model in capturing the long-term dependence in packet-loss processes for communication networks, and hence provides better prediction of performance measures depending on longer-term correlation of errors. For the purpose of illustration, we will use this model to obtain simulation and numerical analysis results, although we emphasize that any generic Markov channel model can be accommodated by the analysis framework presented in Section IV.

A streaming example with unlimited receiver buffer is illustrated in Figure 3. Without loss of generality, we assume the video starts playing out at time 0 and the transmission begins at time $-\triangle$, where $\triangle$ represents the *initial delay*. In the figure, $G(t)$ represents the amount of data received at the client by time t. When $G(t) < p(t)$, jitter occurs. Then the application stops playing out the video and data accumulates in the receiver buffer, until the jitter-recovery buffering scheme decides to resume displaying.

We study three jitter-recovery buffering schemes in this paper:

- Fixed Jitter Buffering Delay (FBD): after a jitter, buffer for a fixed amount of time D_{jit}, and then resume display;
- Fixed Buffered Playout Data (FPD): after a jitter, buffer for a fixed number of packets B_{jit}, and then resume display; and
- Fixed Buffered Playout Time (FPT): after a jitter, buffer the a number of packets that corresponds to a fixed play out duration T_{jit}.

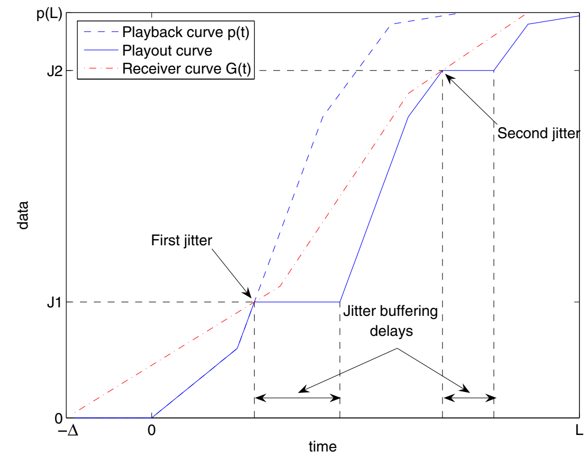


Fig. 3. An example of jitters.

Depending on the channel condition and the jitter-recovery buffering scheme, a client may experience a number of jitters and some jitter buffering delay. Next, we will present our analytical framework and study the tradeoffs among different system parameters.

IV. Performance Analysis of VBR Video Streaming

In this section, we propose a recursive analytical framework to study the effect of initial delay, receiver buffer size, and the parameters of the three aforementioned jitter-recovery buffering schemes. We are particularly interested in computing $E\{N\}$, the expected number of jitters in a given streaming session. The key notation introduced in this section is summarized in Table I for easy reference.

A. Overview of Analysis Framework

We define the position of a jitter by the packet index where it occurs. Thus, by "a jitter occurs at packet i," we mean that the ith packet is the first packet whose playout deadline d_i is violated after the start of display or the previous jitter. Denote J_n as the position of the nth jitter, and X_n as the channel state when the nth jitter occurs. Also, let $P_k^{(n)}(i)$ be the probability that the nth jitter occurs at packet i in channel state S_k, i.e.

$$P_k^{(n)}(i) = \Pr\{J_n = i, X_n = S_k\}, \; k = 1, ..., K. \quad (1)$$

Now, let's first assume we have a means to compute the conditional probability that the $(n+1)$th jitter occurs at packet i in channel state S_k, given it is after the nth jitter that occurred at packet j in channel state S_l. We denote it as

$$Q_{l,k}(j, i) = \Pr\{J_{n+1} = i, X_{n+1} = S_k | J_n = j, X_n = S_l\}. \quad (2)$$

Note that $\sum_i \sum_k Q_{l,k}(j, i)$ does not necessarily equal to one, because there may be no more jitters after the one at packet j.

Notation	Definition
$\triangle$	Initial delay
B	Receiver buffer size
t	Distance in time in the video with respect to the beginning
$p(t)$	The minimum amount of data that has to be received by t
d_i	Arrival deadline of the ith packet, $p^{-1}(i)$
J_n	Index of the first packet whose deadline is violated after $n-1$ jitters
X_n	State of the VBR channel when the nth jitter occurs
L	Length of a video
K	Number of states of the channel
R_m	Maximum number of packets that can be transmitted in one time slot
D_{jit}	Fixed jitter buffering delay in FBD
B_{jit}	Fixed jitter buffered playout data in FPD
T_{jit}	Fixed jitter buffered playout time in FPT
N	Total number of jitters during streaming of a video
$\overline{\pi}[i]$	The ith element of vector $\overline{\pi}$

TABLE I
TABLE OF NOMENCLATURE

Moreover, because of the Markovian behavior of the channel and the schemes being considered only make their decision based on the current state, $Q_{l,k}(j,i)$ only depends on the positions and the channel states, but not n.

By direct application of the total probability theorem [15], the probability that the $(n+1)$th jitter occurs at packet i equals to the sum of probabilities that the nth jitter occurs at packets with $j \leq i$ and the $(n+1)$th jitter occurs at packet i. We have

$$\begin{aligned} P_k^{(n+1)}(i) &= \sum_{j=1}^{i}\sum_{l=1}^{K} \Pr\{J_n = j, X_n = S_l, J_{n+1} = i, \\ &\qquad X_{n+1} = S_k\} \\ &= \sum_{j=1}^{i}\sum_{l=1}^{K} Q_{l,k}(j,i) P_l^{(n)}(j). \end{aligned} \tag{3}$$

Then the probability that there are at least n jitters during the playout of a video is

$$\Pr\{N \geq n\} = \sum_{i=1}^{p(L)} \Pr\{J_n = i\} = \sum_{i=1}^{p(L)}\sum_{k=1}^{K} P_k^{(n)}(i). \tag{4}$$

Since $N \geq 0$, the expected number of jitters can be obtained by

$$\begin{aligned} E\{N\} &= \sum_{n=1}^{\infty} \Pr\{N \geq n\} \\ &= \sum_{n=1}^{\infty}\sum_{i=1}^{p(L)}\sum_{k=1}^{K} P_k^{(n)}(i). \end{aligned} \tag{5}$$

Then, if $P_k^{(1)}(i)$ and $Q_{l,k}(j,i)$ are known, we can recursively compute the above metrics. In the rest of this section, we present the derivation of these probabilities for different jitter-recovery buffering schemes and for both the infinite buffer and finite buffer cases.

B. Infinite Buffer Case

We first consider the case with unlimited receiver buffer, or the buffer size is large enough such that it will not be full during streaming.

1) First Jitter Distribution $P_k^{(1)}(i)$: The first jitter distribution $P_k^{(1)}(i)$ is common to all jitter-recovery buffering schemes, and hence we provide its derivation first. Denote

$$F_{l,k,r} = \Pr\{S_l \rightarrow S_k\} e(r_k, r), \; r \leq R_m, \tag{6}$$

as the probability that, given the channel state in the previous step is S_l, the current channel state is S_k, and r packets are successfully transmitted. Here $e(r_k, r)$ equals to 1 if $r_k = r$ and 0 otherwise, and R_m is the maximum number of packets that can be transmitted in one time slot. We can express the state of the system with the tuple (g, s), where g specifies the number of received packets, and $s \in S$ specifies the channel state. Then we can construct a Markov chain of the system with the following transition probability matrix

$$\mathbf{\Phi} = \begin{array}{c} {\scriptstyle 0} \\ {\scriptstyle 1} \\ \vdots \\ {\scriptstyle p(L)-1} \\ {\scriptstyle p(L)} \\ \vdots \end{array} \begin{bmatrix} \mathbf{A}_0\mathbf{A}_1\cdots\mathbf{A}_{R_m} & 0 & & \cdots & 0 \\ 0 \; \mathbf{A}_0\mathbf{A}_1 & \cdots & \mathbf{A}_{R_m} & 0 \; \cdots & 0 \\ \vdots \; \ddots \; \ddots & \ddots & & \ddots \; \ddots & \vdots \\ 0 \cdots 0 & \mathbf{A}_0 & \mathbf{A}_1 & \cdots \mathbf{A}_{R_m} & 0 \\ 0 \quad \cdots & 0 & \mathbf{A}_0 & \mathbf{A}_1 \; \cdots & \mathbf{A}_{R_m} \\ \vdots & & & \ddots & \end{bmatrix}, \tag{7}$$

where

$$\mathbf{A}_r = \begin{array}{c} \\ {\scriptstyle (g,S_1)} \\ \vdots \\ {\scriptstyle (g,S_K)} \end{array} \begin{array}{c} {\scriptstyle (g+r,S_1) \cdots (g+r,S_K)} \\ \begin{bmatrix} F_{1,1,r} & \cdots & F_{1,K,r} \\ \vdots & \ddots & \vdots \\ F_{K,1,r} & \cdots & F_{K,K,r} \end{bmatrix} \end{array}. \tag{8}$$

With this transition matrix, we can obtain the distribution of the number of packets that have been received by time t simply from $\overline{\pi}_{init}\mathbf{\Phi}^{\triangle+t}$, where $\overline{\pi}_{init}$ is the initial state distribution at $-\triangle$. However, what we are interested in is the probability of the system arriving at a state without any jitter by time t. Moreover, since the video is VBR encoded, the consumption speed of data is predetermined but varies over time. Hence the standard homogeneous Markov chain approach [15] can not be applied in this non-homogeneous case.

Instead, we propose the following. At any time t, states with $g < p(t)$ should not be considered for the computation of the state distribution of the next time slot because these states have already violated the playout deadline by time t. Figure 4 illustrates the idea: the contribution of arrows starting from the shaded states should be eliminated. This can be done by setting $\overline{\pi}_t[Kg+l]$, the $(Kg+l)$th element of $\overline{\pi}$ corresponding the state (g, S_l), to 0 for $g < p(t), \; l = 1, ..., K$, before using it to compute the state distribution of the next time slot. It is equivalent to modifying $\mathbf{\Phi}$ into $\mathbf{\Phi}\mathbf{U}_t$, with

$$\mathbf{U}_t = \begin{bmatrix} \mathbf{0}_{Kp(t)\times Kp(t)} & 0 \\ 0 & \mathbf{I} \end{bmatrix}, \tag{9}$$

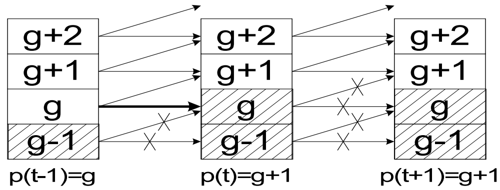


Fig. 4. Arrows indicate the transition of states. The shaded blocks represent the states that have violated the playout deadline. The thick arrow indicates the transition that introduces a jitter at packet g at time t.

in each time slot. Then the probability of arriving at a state without having jitter by time $d_i - 1$ is given by

$$\overline{\pi}_{d_i-1} = \overline{\pi}_{init}(\prod_{t=-\triangle}^{d_i-1} \mathbf{\Phi}\mathbf{U}_t). \tag{10}$$

And the value of the $(K(i-1)+k)$th $(k \leq K)$ element of

$$\overline{\pi}'_{d_i} = \overline{\pi}_{d_i-1}\mathbf{\Phi} = \overline{\pi}_{init}(\prod_{t=-\triangle}^{d_i-1} \mathbf{\Phi}\mathbf{U}_t)\mathbf{\Phi} \tag{11}$$

is the probability that there has been no jitter by time $d_i - 1$, and the client has only received $i-1$ packets in channel state S_k by d_i, which is $P_k^{(1)}(i)$, i.e.,

$$P_k^{(1)}(i) = \overline{\pi}'_{d_i}[K(i-1)+k]. \tag{12}$$

2) Next Jitter Probability $Q_{l,k}(j,i)$: Next, we will study the distributions of the position of the subsequent jitter for the aforementioned jitter recovery buffering schemes.

a) Fixed Jitter Buffering Delay: In the FBD scheme, every time a jitter occurs, the application stops displaying and buffers data for a fixed jitter buffering delay D_{jit}. The procedure to find $Q_{l,k}(j,i)$ is similar to that of finding the first jitter probability in section IV-B1. We only need to imagine we start playing out the video from packet j with an empty buffer. Then $(j-1, S_l)$ is the virtual "initial" state, and D_{jit} is the virtual "initial" delay. Denote the virtual initial state distribution after a jitter occurs at (j, S_l) as

$$\overline{\pi}_{j,l} = [0 \ \cdots \ 0 \quad \underset{\text{the } (K(j-1)+l) \text{ th element}}{1} \quad 0 \ \cdots \ 0]. \tag{13}$$

Then similar to equation (11), the state probability distribution at time d_i of having no jitter by $d_i - 1$ is given by

$$\overline{\pi}'_{d_i} = \overline{\pi}_{j,l}(\prod_{t=d_j-D}^{d_i-1} \mathbf{\Phi}\mathbf{U}_t)\mathbf{\Phi}. \tag{14}$$

And $Q_{l,k}(j,i)$ is obtained by

$$Q_{l,k}(j,i) = \overline{\pi}'_{d_i}[K(i-1)+k]. \tag{15}$$

b) Fixed Buffered Playout Data: In the FPD scheme, after a jitter happens, the application stops playing out the video and buffers data until the number of packets in the buffer reaches B_{jit}. If a jitter occurs at the jth packet, it will restart playing out the video once the $(j + B_{jit} - 1)$th packet arrives. To find $Q_{l,k}(j,i)$, we first have to find the probability distribution of the state (g, S_h) where playout is restarted, where $g \in [j+B_{jit}-1, j+B_{jit}+R_m-2]$ represents the last packet received when display resumes. To find these probabilities, we construct a Markov chain of the receiver buffer during this buffering stage with transition probability matrix

$$\mathbf{\Psi} = \begin{array}{c} {\scriptstyle j-1} \\ {\scriptstyle j} \\ \vdots \\ {\scriptstyle j+B_{jit}-2} \\ {\scriptstyle j+B_{jit}-1} \\ \vdots \\ {\scriptstyle j+B_{jit}+R_m-2} \end{array} \begin{bmatrix} \mathbf{A}_0\mathbf{A}_1 \cdots \mathbf{A}_{R_m} & 0 & \cdots & 0 & \\ 0 \ \mathbf{A}_0\mathbf{A}_1 & \cdots & \mathbf{A}_{R_m} & 0 & \cdots \\ \vdots \ \ddots \ \ddots & & & & \vdots \\ 0 \cdots 0 & \mathbf{A}_0 & \mathbf{A}_1 & \cdots \mathbf{A}_{R_m} & \\ 0 \cdots & 0 & \mathbf{I} & 0 & \cdots \\ \vdots & \cdots & \ddots & \ddots & 0 \\ 0 & & \cdots & 0 & \mathbf{I} \end{bmatrix}. \tag{16}$$

States with $g \in [j + B_{jit} - 1, j + B_{jit} + R_m - 2]$ are modeled as absorption states, because once the system arrives at any one of these states, it exits the jitter recover buffering stage, and resumes playing out the video. Then the distribution of resuming display among these states can be obtained by solving the absorption probabilities of these states, starting from state $(j-1, S_l)$ [15].

Now we can compute $Q_{l,k}(j,i)$ for the FPD scheme. Let $\overline{\pi}_{j,l}$ be the state distribution when exiting the buffering stage after a jitter at (j, S_l), where each element indexed by $Kg+h$, for $g = j + B_{jit} - 1, ..., j + B_{jit} + R_m - 2$ and $h \leq K$, equals the absorption probability of the corresponding state (g, S_h). Noting that, after exiting the buffering stage, the system restarts playing out the video from packet j, we can compute $Q_{l,k}(j,i)$ from

$$\overline{\pi}'_{d_i} = \overline{\pi}_{j,l}(\prod_{t=d_j}^{d_i-1} \mathbf{\Phi}\mathbf{U}_t)\mathbf{\Phi}, \tag{17}$$

and

$$Q_{l,k}(j,i) = \overline{\pi}'_{j,l}[K(i-1)+k], \ i \geq j + B_{jit}. \tag{18}$$

Moreover, in order to compare FPD with the FBD scheme, we want to quantify the jitter recover buffering delay of FPD. Since the delay is not constant, we will look into the expected buffering delay. The expected buffering delay after a jitter at (j, S_l) can be obtain by solving the mean time to absorption of the above Markov chain [15]. Let m_k be the mean time to absorption into any one of the absorption states, starting from state S_k, which can be solved from the probability transition matrix $\mathbf{\Psi}$. Then the expected buffering delay of the nth jitter is

$$E\{D_n\} = \sum_{i}^{p(L)} \sum_{k=1}^{K} P_k^{(n)}(i) m_k. \tag{19}$$

Here we define $D_n = 0$ if the nth jitter does not exist. And the expected total jitter buffering delay experienced by the client becomes

$$E\{D_{total}\} = \sum_{n=1}^{\infty} E\{D_n\} = \sum_{n=1}^{\infty} \sum_{i=1}^{p(L)} \sum_{k=1}^{K} P_k^{(n)}(i) m_k. \tag{20}$$

Then the average buffering delay per jitter can be approximated by

$$\frac{E\{D_{total}\}}{E\{N\}} = \frac{\sum_{n=1}^{\infty}\sum_{i=1}^{p(L)}\sum_{k=1}^{K} P_k^{(n)}(i)m_k}{\sum_{n=1}^{\infty}\sum_{i=1}^{p(L)}\sum_{k=1}^{K} P_k^{(n)}(i)}. \tag{21}$$

c) Fixed Buffered Playout Time: The FPT scheme is similar to the FPD scheme, except that, rather than buffering a fixed amount of data before resuming display, the FPT scheme buffers data until the playout duration of these data reaches a fixed value T_{jit}. One advantage of this scheme is that, when display resumes, it is guaranteed that the next jitter is at least T_{jit} away in the future. This scheme is used in Windows Media Player and Real Player.

The process to find $Q_{l,k}(j,i)$ is similar to the one in the FPD scheme. The difference is that, for different values of the jitter position j, the number of packets to buffer is $p(d_j + T_{jit}) - (j-1)$, which is not a fixed number as it is in the FPD scheme. Then for each jitter position j, we have a Markov chain characterized by the transition probability matrix

$$\mathbf{\Psi}_j = \begin{array}{r} {\scriptstyle j-1} \\ {\scriptstyle j} \\ \vdots \\ {\scriptstyle p(d_j+T_{jit})-1} \\ {\scriptstyle p(d_j+T_{jit})} \\ \vdots \\ {\scriptstyle p(d_j+T_{jit})+R_m-1} \end{array} \begin{bmatrix} \mathbf{A}_0\mathbf{A}_1\cdots\mathbf{A}_{R_m} & 0 & \cdots & 0 & & \\ 0 & \mathbf{A}_0\mathbf{A}_1 & \cdots & \mathbf{A}_{R_m} & 0 & \cdots \\ \vdots & \ddots & \ddots & & & \vdots \\ 0\cdots 0 & \mathbf{A}_0 & \mathbf{A}_1 & \cdots & \mathbf{A}_{R_m} & \\ 0\cdots & 0 & \mathbf{I} & 0 & \cdots & \\ \vdots & & \cdots & \ddots & \ddots & 0 \\ 0 & & & \cdots & 0 & \mathbf{I} \end{bmatrix} \tag{22}$$

Then we can compute the state distribution when exiting the buffering stage after a jitter at (j, S_l), $\overline{\pi}_{j,l}$, by computing the absorption probabilities of $\mathbf{\Psi}_j$ starting from state (j, S_l). Similar to (17) and (18), we have

$$\overline{\pi}'_{d_i} = \overline{\pi}_{j,l}\Big(\prod_{t=d_j}^{d_i-1}\mathbf{\Phi}\mathbf{U}_t\Big)\mathbf{\Phi}, \tag{23}$$

and

$$Q_{l,k}(j,i) = \overline{\pi}'_{j,l}[K(i-1)+k]. \tag{24}$$

Further, for a different $\mathbf{\Psi}_j$, the mean time to absorption is different. We use $m_k(i)$ to denote the mean time to absorption of $\mathbf{\Psi}_i$, starting from $(i-1, S_k)$, which is obtained by solving $\mathbf{\Psi}_i$. Then similar to the FPD scheme, the average buffering delay per jitter is approximated by

$$\frac{E\{D_{total}\}}{E\{N\}} = \frac{\sum_{n=1}^{\infty}\sum_{i=1}^{p(L)}\sum_{k=1}^{K} P_k^{(n)}(i)m_k(i)}{\sum_{n=1}^{\infty}\sum_{i=1}^{p(L)}\sum_{k=1}^{K} P_k^{(n)}(i)}. \tag{25}$$

C. Finite Buffer Case

In the previous sections, we have studied the case where the receiver buffer size is infinite, or large enough, such that the amount of data received is not limited from above. Next, we study the case where the receiver buffer size is limited.

Suppose the receiver buffer size is B. At any video position t, we can have at most received $p(t)+B$ packets. We consider systems in which the client continuously updates its buffer fullness to the server, the server stops sending data packets when it knows the receiver buffer is full, and the server restarts sending data packets once the buffer starts to empty again[1]. Figure 5 illustrates the state transitions in this case. The transitions to any state with $g \geq p(t)+B$ should be merged into the states with $g = p(t)+B-1$. This can be accomplished by setting

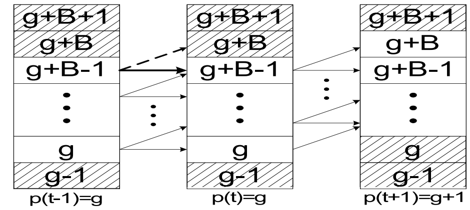


Fig. 5. Solid arrows indicate the transition of states that should be considered. The shaded blocks indicate the states that have violated the playout deadline or that have surpassed the receiver buffer limit. The dotted arrow represents the state transitions that should be merged into the state transitions represented by the thick solid arrow.

$$\overline{\pi}_t[K(p(t)+B-1)+l] = \sum_{g\geq p(t)+B-1}\overline{\pi}_t[Kg+l], \tag{26}$$

and $\overline{\pi}_t[Kg+l] = 0$ for $g \geq p(t)+B$ and $l \leq K$, while computing the state distribution. And this operation can be formulated in matrix multiplications similar to (10) as

$$\overline{\pi}_{d_i-1} = \overline{\pi}_{init}\Big(\prod_{t=-\triangle}^{d_i-1}\mathbf{\Phi}\mathbf{U}'_t\Big), \tag{27}$$

where

$$\mathbf{U}'_t = \begin{array}{c} \begin{array}{cccc} \phantom{\mathbf{0}_{Kp(t)\times Kp(t)}} & \phantom{\mathbf{I}_{K(B-1)\times K(B-1)}} & {\scriptstyle p(t)+B-1} & \end{array} \\ \begin{bmatrix} \mathbf{0}_{Kp(t)\times Kp(t)} & 0 & 0 & 0 \\ 0 & \mathbf{I}_{K(B-1)\times K(B-1)} & 0 & 0 \\ 0 & 0 & \mathbf{I}' & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \end{array}, \tag{28}$$

and

$$\mathbf{I}' = [\underbrace{\mathbf{I}_{K\times K} \;\cdots\; \mathbf{I}_{K\times K}}_{R_m+1}]^T. \tag{29}$$

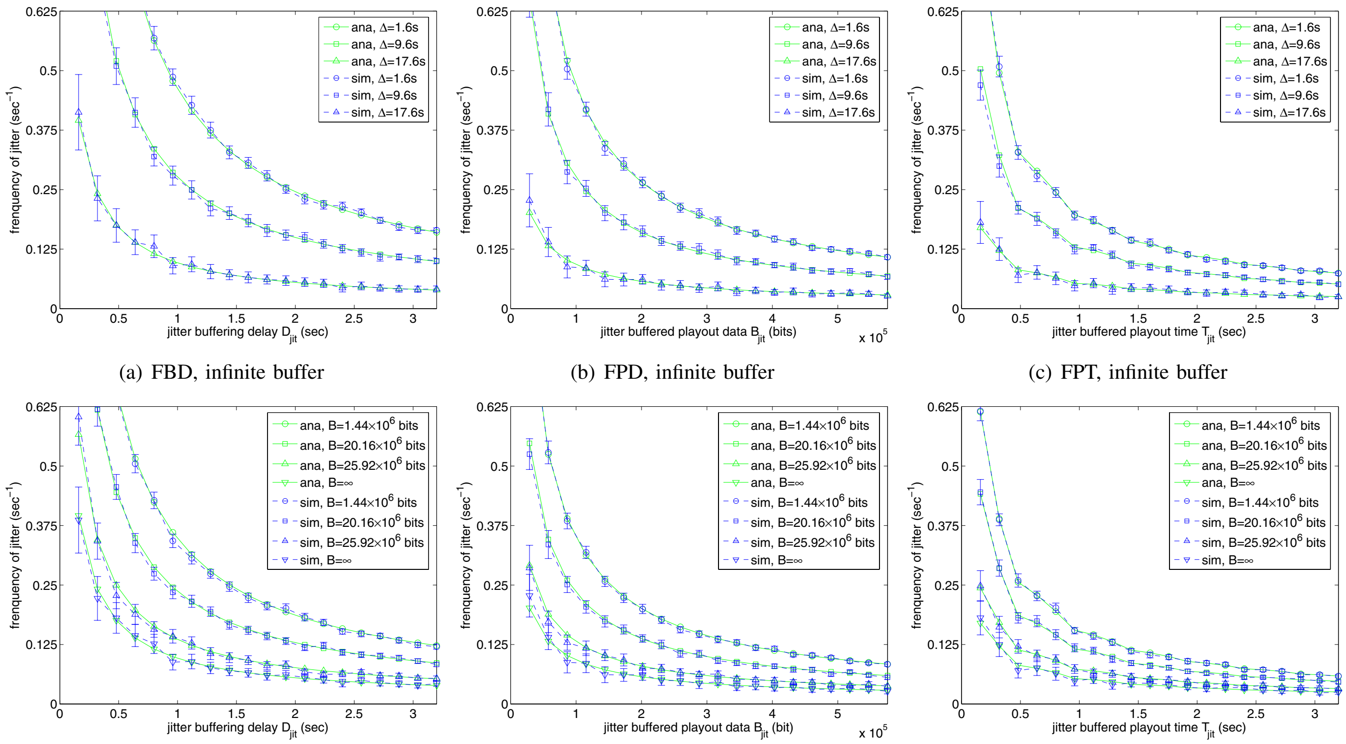
$\mathbf{I}'$ here is a $K(R+1)\times K$ matrix. By doing this, any state with $g \geq p(t)+B$ is merged into a state with $g = p(t)+B-1$ according to its channel state. Then (11) can be rewritten as

$$\overline{\pi}'_{d_i} = \overline{\pi}_{init}\Big(\prod_{t=-\triangle}^{d_i-1}\mathbf{\Phi}\mathbf{U}'_t\Big)\mathbf{\Phi}. \tag{30}$$

The process is similar while computing $Q_{l,k}(j,i)$. We only need to replace $\mathbf{U}_t$ with $\mathbf{U}'_t$ in (17) and (23), and in (14) for $t \geq d_j$. In (14) when $t < d_j$, we should set

$$\mathbf{U}'_t = \begin{array}{c} \begin{array}{cccc} \phantom{\mathbf{0}_{Kj\times Kj}} & \phantom{\mathbf{I}_{K(B-1)\times K(B-1)}} & {\scriptstyle j+B-1} & \end{array} \\ \begin{bmatrix} \mathbf{0}_{Kj\times Kj} & 0 & 0 & 0 \\ 0 & \mathbf{I}_{K(B-1)\times K(B-1)} & 0 & 0 \\ 0 & 0 & \mathbf{I}' & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \end{array}, \tag{31}$$

[1]This mechanism is vital, as jitters due to buffer overflow would occur without it.



(a) FBD, infinite buffer (b) FPD, infinite buffer (c) FPT, infinite buffer

(d) FBD, finite buffer, $\triangle = 17.6s$ (e) FPD, finite buffer, $\triangle = 17.6s$ (f) FPT, finite buffer, $\triangle = 17.6s$

Fig. 6. Frequency of jitters vs. D_{jit}, B_{jit} and T_{jit} for FBD, FPD and FPT schemes, respectively, for video "Alpin Ski".

because within the buffering delay, the buffer can contain up to the $(j+B-1)$th packet.

V. Experiment and Numerical Results

In this section, we apply the proposed analysis framework to study how different jitter recovering schemes and the choice of parameters affect the jitter performance in VBR video streaming. The analysis results are validated by comparison with simulation results. We further investigate the optimal balance between jitters and buffering delay.

A. Experimental Setup

We experiment on wireless streaming using MPEG-4 VBR video traces provided by [16]. Some main statistics of these videos clips are listed in Table II. These sequences were encoded at a constant frame rate of 25 frames/s in the Quarter Common Intermediate Format (QCIF) resolution (176×144). We have chosen the QCIF format because we are particularly interested in wireless networking systems where hand-held wireless devices typically have a screen size that corresponds to the QCIF video format.

To emulate a realistic wireless channel, we adopt a three-state extended Gilbert model shown in Figure 2 as the VBR channel model (i.e., $k = 2$), with transition probabilities $\Pr\{0|1\} = 0.1$, $\Pr\{0|10\} = 0.5$ and $\Pr\{0|00\} = 0.8$. Then the transition matrixes are constructed by

$$A_0 = \begin{bmatrix} 0 & 0.1 & 0 \\ 0 & 0 & 0.5 \\ 0 & 0 & 0.8 \end{bmatrix}, \text{ and } A_1 = \begin{bmatrix} 0.9 & 0 & 0 \\ 0.5 & 0 & 0 \\ 0.2 & 0 & 0 \end{bmatrix}.$$

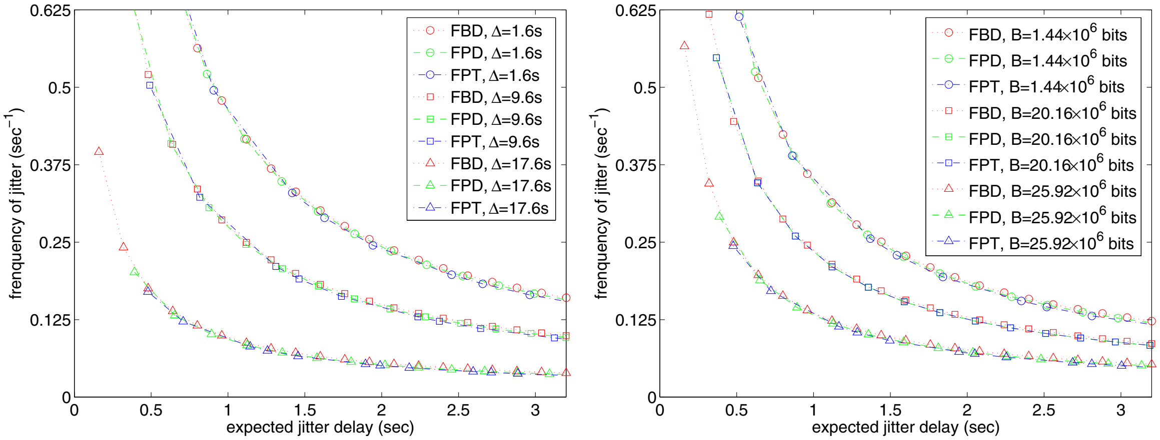
Video name	Parameters of the video stream	
	Mean bit rate (per sec)	Peak bit rate (per sec)
Alpin Ski	1.8247e+05	3.0248e+06
Formula 1	1.7754e+05	2.1428e+06
Jurassic	1.7100e+05	3.1668e+06

TABLE II
Parameters of different video streams

We set the packet size to 1800 bytes and the time interval between two consecutive transmission to 80 ms. This results in a maximum transmission bit rate of 180,000 bit/s. For different initial delay values, receiver buffer sizes and jitter recover scheme parameters, we simulate in Matlab the transmission and playback of each video sequence over 200 realizations of the random VBR channel and measure the number of jitters and the buffering delay of each jitter. Furthermore, we compute the expected number of jitters and expected buffering delay based on the proposed analysis and compare them with the simulation results.

B. Numerical Validation

In Figure 6, we plot the frequency of jitters vs. jitter recovery parameters for video "Alpin Ski" in different initial delay and receiver buffer size settings. Figure 6(a) and 6(d) are based on the fixed jitter buffering delay FBD scheme as in IV-B2a. Figure 6(b) and 6(e) are based on the fixed jitter buffered playout data FPD scheme as in IV-B2b. Figure 6(c) and 6(f) are based on the FPT scheme as in IV-B2c. The 95% confidence intervals for the simulations are also shown in these figures. We observe good match between the derived



(a) Infinite buffer with different initial delays

(b) Finite buffers with fixed initial delay $\triangle = 17.6s$

Fig. 7. Jitter frequency vs. average jitter recovery buffering delay, for three jitter recovery buffering schemes.

expected number of jitters and simulation. Similar results are observed for different playback curves and omitted to reduce redundancy.

It can be seen in Figure 6(a) that, for all initial delay and buffer size settings, there is deminishing gain by increasing D_{jit} on the performance: the number of jitters decreases dramatically as D_{jit} increases at the beginning and then slowly after D_{jit} surpasses a certain value. We also observe that for a fixed buffer size, a larger initial delay results in smaller number of jitters and a more dramatic decrease as D_{jit} increases. On the other hand, for a fixed initial delay, a larger buffer size also results in fewer jitters and a sharper drop.

The practical implication of this observation is clear. The jitter buffering delay is a delicate parameter in the optimal design of multimedia streaming. Increasing the jitter buffering delay can drastically reduce the number of jitters, but only up to a certain level. Beyond that, the improvement is negligible, and the long jitter buffer delay may actually harm the perceived quality of the streaming.

We further note that similar observations can be found in other jitter-recovery buffering schemes.

C. Comparison of Jitter-Recovery Buffering Schemes

In Figure 7 we plot the jitter frequency vs. expected buffering delay under the three jitter recovery schemes, with different fixed initial delays and receiver buffer sizes. For the FBD scheme, the expected delay is just D_{jit}. For the FPD and FPT schemes, the expected delays are given by equation (21) and (25). It is interesting to note that these schemes provide very similar tradeoff between jitter frequency and the expected jitter recovery buffering delay.

However, as shown in Figure 8, the FBD scheme has zero delay variance because the delay is fixed, while the other two schemes have large delay variances. Moreover, the FPT scheme has much larger delay variance than the FPD scheme. This is because in the FPT scheme, the amount of data that has to be buffered varies drastically when jitter occurs at different position in the video, which results in more fluctuations, in comparison with the FPD scheme. In another words, while

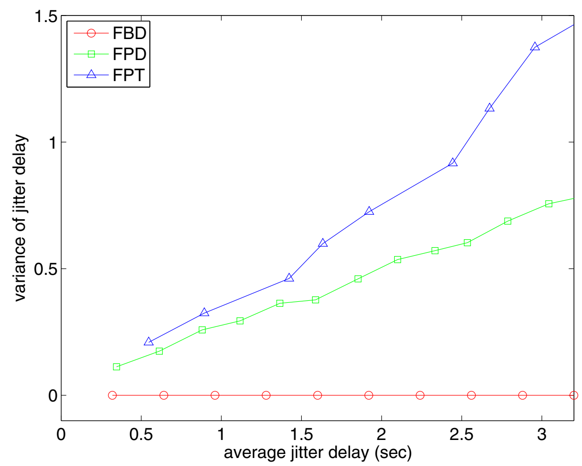


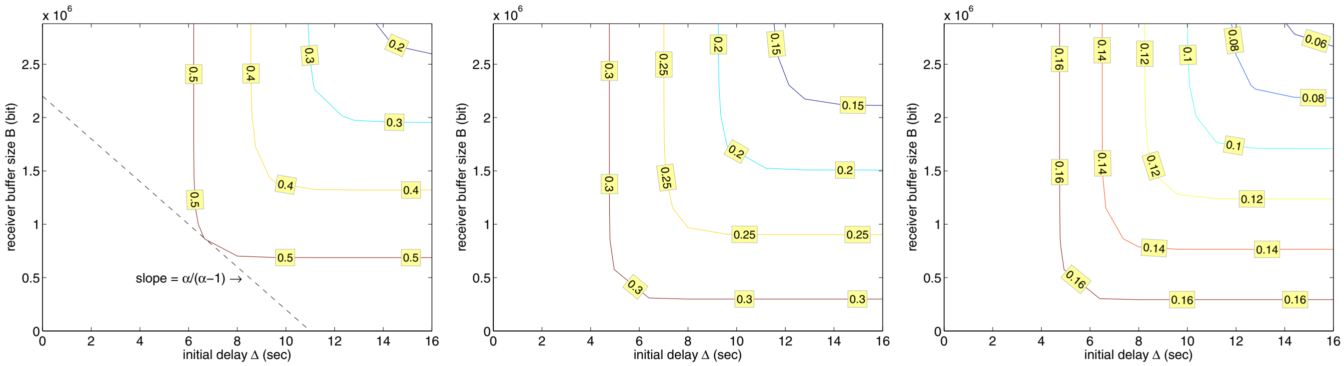
Fig. 8. Variance of jitter recovery buffering delay.

jitters can not occur close to each other in the FPT scheme, they are more likely to incur unacceptably long buffering delay compared with the other two schemes.

D. Optimal Delay-Buffer Tradeoff

Finally, we consider the optimal tradeoff between the initial playback delay $\triangle$ and the receiver buffer size B in video streaming. In most cases, long buffering delay in the middle of a streaming session is less acceptable for a streaming client. Therefore, a client may impose stiff constraints on the jitter recovery buffering delay and may wish to tradeoff the initial delay and the receiver buffer size for a lower frequency of jitters. Figure 9 shows contour maps for the initial delay and the buffer size that achieve different frequencies of jitters, for different fixed jitter recover buffering delays.

We observe that the right branch of a contour curve is roughly horizontal. This suggests that when the buffer size is fixed, increasing the initial delay can only decrease the frequency of jitters to a certain level. When $\triangle$ is large enough, the buffer is always filled up during the initial delay, and the display of the video always begins with a full buffer. Further increasing the initial delay will not change this situation,



(a) $D_{jit} = 0.8s$ (b) $D_{jit} = 1.6s$ (c) $D_{jit} = 3.2s$

Fig. 9. Contour maps of buffer size vs. initial delay, with the frequency of jitters labeled on the curves.

resulting in no improvement of performance. Similarly, the left branch of a contour curve is roughly vertical. The rational behind this is that, when B is large enough, the receiver buffer will never be completely filled throughout a streaming session, which is equivalent to the case with infinite buffer. Finally, these figures provide a convenient means to obtain an optimal operating point for the system that balances the tradeoff between $\triangle$ and B, given a certain required frequency of jitters and jitter recovery buffering delay. If we define a cost function as a weighted sum of the two

$$C = \alpha\triangle + (1-\alpha)B, \tag{32}$$

where $\alpha \in [0,1]$, then to minimize this cost function, we simply find the tangent line of the corresponding contour curve with slope $\frac{\alpha}{\alpha-1}$. The point of contact between the tangent line and the contour curve, given any value of frequency of jitter, defines the optimal operating point for the system. An illustration of such a procedure is shown in Figure 9(a).

VI. Conclusions

In this work, we have considered the problem of providing QoS to VBR encoded video streaming service over random VBR channels. We have shown that, when some statistical characteristics of the channel, such as the channel state transition probability, are available, a certain level of QoS can be guaranteed by selecting appropriate jitter recovery schemes.

We present an analytical framework that only requires knowledge of the playback curve and a Markov channel model. The frequency of jitters and the expected jitter recovery buffering delay have been derived for both the infinite buffer and finite buffer cases. Numerical and experimental results using MPEG-4 encoded VBR video traces validate our findings. The proposed numerical analysis allow precise estimation of the effect of the choice of jitter recovery schemes, initial playout delay and receiver buffer size. We show that the FBD, FPD, and FPT schemes provide similar tradeoff performance between the jitter frequency and the jitter recovery buffering delay, while the FBD scheme incurs zero buffering delay variance and the FPT scheme incurs the worse buffering delay variance. To practical streaming system designers, the proposed analysis technique provides a convenient framework to optimize the tradeoffs between the various system parameters for optimal VBR multimedia streaming over random VBR channels.

References

[1] M. Etoh, and T. Yoshimura, "Advances in Wireless Video Delivery," *Proc. IEEE*, vol. 93, no. 1, pp. 111–122, Jan 2005.

[2] C. Chiasserini, M. Meo, D. Tarfanelli, and D. Visconti, "A Study of Video Services in a Wireless Environment," in *The 4th IEEE Conference on Mobile and Wireless Communitions Networks*, Sept 2002.

[3] T. V. Lakshman, A. Ortega, and A.R. Reibman, "VBR Video: Trade-offs and Potentials," *Proc. IEEE*, vol. 86, no. 1, pp. 952–973, May 1998.

[4] S. Sen, J. L. Rexford, J. K. Dey, J. F. Kurose, and D. F. Towsley, "Online Smoothing of Variable-Bit-Rate Streaming Video," *IEEE Trans. Multimedia*, vol. 2, no. 1, pp. 37–48, Mar 2000.

[5] P. Thiran, J.-Y. Le Boudec, and F Worm,, "Network Calculus Applied to Optimal Multimedia Smoothing," in *Proc. IEEE Infocom*, Apr 2001.

[6] V. Varsa and I. Curcio, "Transparent End-to-End Packet Switched Streaming Service (PSS); RTP Usage Model (Release 5)," *3GPP TR 26.937 V1.4.0*, 2003.

[7] T. Stockhammer, H. Jenkac, and G. Kuhn, "Streaming Wideo over Variable Bit-Rate Wireless Channels," *IEEE Trans. Multimedia*, vol. 6, no. 2, pp. 268–277, Apr 2002.

[8] M. Kalman, E. Steinbach, and B. Girod, "Adaptive Media Playout for Low-Delay Video Streaming Over Error-Prone Channels," *IEEE Trans. Circuits and System for Video Technology*, vol. 14, no. 6, pp. 841–851, Jun 2004.

[9] E. N. Gilbert, "Capacity of a Burst-Noise Channel," *Bell Syst. Tech. J.*, vol. 39, no. 5, pp. 1253–1265, Oct 1960.

[10] E. O. Elliott, "Estimates of Error Rates for Codes on Burst-Noise Channels," *Bell Syst. Tech. J.*, vol. 42, pp. 1977–1997, 1963.

[11] L. Xu and J. Helzer, "Media Streaming via TFRC: An Analytical Study of the Impact of TFRC on User-Perceived Media Quality," in *Proc. IEEE INFOCOM 2006*, Apr 2006.

[12] E. Biersack, A. Jean-Marie, and P. Nain, "Open-Loop Video Distribution with Support of VCR Functionality," *Performance Evaluation*, vol. 49, no. 1-4, pp. 411–427, 2002.

[13] Guanfeng Liang and Ben Liang, "Jitter Free Probability Bounds for Video Streaming over Random VBR Channel," in *Proc. the Third International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks*, Aug 2006.

[14] H. Sanneck, G. Carle, and R. Koodli, "A Framework Model for Packet Loss Metrics Based on Loss Run Length," in *Proceedings of SPIE/ACM SIGMM Multimedia Computing and Networking Conference*, Jan 2000.

[15] A. Papoulis and S. U. Pillai, *Probability, Random Variables, and Stochastic Processes*, Fourth ed. McGraw-Hill, 2002.

[16] F.H.P. Fitzek and M. Reisslein, "MPEG-4 and H.263 Video Traces for Network Performance Evaluation," *IEEE Network*, vol. 15, no. 6, pp. 40–54, November 2001.