Lexicographic Bit Allocation for MPEG Video

Dzung T. Hoang

Sony Semiconductor of America

3300 Zanker Road, MS SJ3C3

San Jose, CA 95134

dth@ricochet.net

Elliot L. Linzer

C-Cube Microsystems

One Water Street, 2nd Floor

White Plains, NY 10601

elliot.linzer@c-cube.com

Jeffrey Scott Vitter

Duke University

Box 90129

Durham, NC 27708–0129

jsv@cs.duke.edu

Proposed Running Head: Lexicographic Bit Allocation for MPEG Video

Corresponding Author: Dzung T. Hoang

Sony Semiconductor of America

3300 Zanker Road, MS SJ3C3

San Jose, CA 95134

E-mail: dth@ricochet.net

Phone: 408–666–3991

Abstract

We consider the problem of allocating bits among pictures in an MPEG video coder to equalize the visual quality of the coded pictures, while meeting buffer and channel constraints imposed by the MPEG Video Buffering Verifier. We address this problem within a framework that consists of three components: 1) a bit production model for the input pictures, 2) a set of bit-rate constraints imposed by the Video Buffering Verifier, and 3) a novel lexicographic criterion for optimality. Under this framework, we derive simple necessary and sufficient conditions for optimality that lead to efficient algorithms.

Table of Typographical Symbols

Symbol	Name
α	Greek alpha
β	Greek beta
γ	Greek gamma
∞	infinity
$\emptyset$	empty set
$\asymp$	LaTeX `\asymp`
$\succ$	LaTeX `\succ`
$\succeq$	LaTeX `\succeq`
$\prec$	LaTeX `\prec`
$\preceq$	LaTeX `\preceq`

1 Introduction

In any lossy coding system, there is an inherent trade-off between the rate of the coded data and the distortion of the reconstructed signal. Often the transmission (storage) medium is bandwidth (capacity) limited. The purpose of rate control is to allocate bits to coding units and to regulate the coding rate to meet the bit-rate constraints imposed by the transmission or storage medium while maintaining an acceptable level of distortion.

We consider rate control in the context of the MPEG-1 and MPEG-2 video coding standards. In addition to specifying a syntax for the encoded bitstream and a mechanism for decoding it, the MPEG standards define a hypothetical decoder called the Video Buffering Verifier (VBV), which places quantifiable limits on the variability in bit rate of encoded video. The VBV is an integral part of the MPEG standards and MPEG-compliant bitstreams must be decodable by the VBV.

In this paper, we develop a novel framework for bit allocation under VBV constraints and a total bit budget. This framework consists of three components: 1) a bit-production model, 2) a novel lexicographic optimality criterion, and 3) a set of buffer constraints for constant and variable bit rate operation. We formalize bit allocation as a resource allocation problem with continuous variables and non-linear constraints, to which we apply a global lexicographic optimality criterion.

The goal of optimal bit allocation has traditionally been to minimize an additive distortion measure, typically mean-squared error (MSE), averaged over coding blocks. While this approach leverages the wealth of tools from optimization theory and operations research, it does not guarantee the constancy in quality that is generally desired from a video coding system. For example, a video sequence with a constant or near-constant level of distortion is more desirable than one with lower average distortion but higher variability, because human viewers tend to find frequent changes in quality more noticeable and annoying. A long video sequence typically contains segments that, even if encoded at a fairly low bit rate, will not contain any disturbing quantization artifacts, so that improving the quality of pictures in those segments is far less important than improving the quality of pictures in segments that are more difficult to encode.

To address these issues, we propose a *lexicographic optimality* criterion that better expresses the desired constancy in quality. The idea is to minimize the maximum (perceptual) distortion of a block (or picture) and then minimize the second highest block distortion, and so on. The intuition is that doing so would equalize distortion by limiting peaks in distortion to their minimum. As we will show later, if a constant quality allocation is feasible, then it must necessarily be lexicographically optimal.

The global nature of lexicographic optimization necessitates the use of off-line techniques wherein the complexities of all the coded pictures, as specified with bit-production models, are known prior to computing a global bit allocation. One way to view this is as a serial computation with unlimited lookahead, wherein the inputs are the bit production models for each picture. In practice, this would entail making multiple passes over the video sequence in order to construct the models, compute an optimal allocation, and compress the sequence using the computed allocation. In Section 8, we explore some techniques for reducing the computation by limiting the amount of lookahead used.

In Section 3, we detail our new lexicographic framework for bit allocation. In Sections 3.3.1 and 3.3.2, we analyze bit allocation with constant-bit-rate and variable-bit-rate constraints. The analyses yield necessary and sufficient conditions for optimality that lead to efficient bit-allocation algorithms. In Section 8, we describe an implementation of these algorithms within a software MPEG-2 encoder and present simulation results.

2 Previous Work

Shoham and Gersho [1] have examined the budget-constrained bit-allocation problem in the context of a discrete set of independent quantizers. A bit-allocation algorithm based upon Lagrangian minimization is presented as a more efficient alternative to a well-known dynamic programming solution based upon the Viterbi Algorithm [2, 3]. Although it only solves the simple budget-constrained allocation problem, this work lays the foundation for much of the ensuing work on optimal bit allocation.

Optimal budget-constrained bit allocation in a dependent-coding setting is examined in [4]. A parametric rate-distortion model is proposed for intraframe coding and forward predictive coding. The model has an exponential form and is motivated by theoretical rate-distortion results for stationary Gaussian sources. Lagrangian minimization is chosen as the optimization technique and a closed-form solution is obtained in terms of known statistics and the Lagrange multiplier. A search over the Lagrange multiplier then yields a solution to the budget-constrained problem. The authors acknowledge that minimizing sum-distortion does not lead to uniform distortion. They reformulate the problem to minimize the maximum (minimax) picture distortion. However, they do not refine the minimax solutions to further minimize the second highest distortion, and so on. The minimax solution is obtained by equating the distortion among pictures.

Budget-constrained minimax bit allocation for dependent coding is also considered in [5]. The authors provide a minimax solution by first showing how to find a minimum-rate solution given a maximum distortion and then using a bisection search to find the maximum distortion corresponding to the desired rate. However, the bisection search is not guaranteed to converge in a finite number of iterations.

The problem of optimal bit allocation in a buffered video coder is first presented in [6]. The authors consider video coding with CBR buffer constraints and formulate bit allocation as an integer-programming problem. They assume a finite set of quantization scales, an integral number of coded bits, and independent coding. The problem is optimally solved using a dynamic programming algorithm based upon the Viterbi Algorithm. Heuristic methods based upon Lagrangian minimization and other ad-hoc techniques are proposed to provide more efficient, but sub-optimal, solutions.

The discrete optimization framework of [6] is extended in [7] to handle dependent coding. Except for a simple illustrative case, computing an optimal bit allocation under the dependent framework requires time and space exponential in the number of coding units. A heuristic pruning technique is proposed to reduce the number of states considered. However, the effectiveness of the heuristic depends upon the rate-distortion characteristics of the source.

The work in [6] is further extended in [8] to include transmission over a variable-bit-rate channel with delay constraints. Besides buffer and delay constraints, the authors also consider constraints imposed by several policing mechanisms proposed for ATM networks. Assuming a discrete set of quantizers and a discrete set of transmission rates, the quantization and transmission rate can be jointly optimized using the Viterbi Algorithm to produce a minimum sum-distortion encoding. In the construction of the trellis used by the Viterbi Algorithm, states that violate the various constraints are discarded. Unlike our framework, there is no explicit constraint on the total number of bits used.

Joint control of encoder and channel rate is also considered in [9]. Instead of considering global optimality, this work focuses on real-time control algorithms. An algorithm is proposed that separates rate control into a "short-term" process and a "long-term" process. The long-term rate control sets a base quantization scale Q_{s} called the *sequence quantization parameter*. In normal operation, Q_{s} is used to code each picture. Long-term rate control monitors the average fullness of a virtual encoder buffer and adjusts Q_{s} to maintain the buffer fullness between two thresholds. Short-term rate control is applied when the upper bound on encoder rate needs to be enforced. Several methods are proposed for performing short-term rate control.

In [10], a model relating bits, distortion, and quantization scale is derived for block-transform video coders. Assuming a stationary Gaussian process, the authors derive a bit-production model containing transcendental functions. The model is applied to control the frame rate of motion-JPEG and H.261 video coders.

In the operations research literature, lexicographic optimality has been applied to such problems as resource location and allocation (e.g., [11, 12, 13, 14, 15, 16]) and is sometimes referred to as *lexicographic minimax*, since it can be viewed as a refinement of minimax theory.

3 Lexicographic Framework

3.1 Perceptual Quantization

The output bit rate of a typical video coder can be regulated by adjusting a quantization scale Q_{s}. Increasing Q_{s} reduces the output bit rate but also decreases the visual quality of the compressed pictures. Similarly, decreasing Q_{s} increases the output bit rate and increases the picture quality.

Although Q_{s} can be used to control rate and distortion, coding with a constant value of Q_{s} generally does not result in either constant bit rate or constant perceived quality. Both of these factors depend upon the scene content as well. Studies into human visual perception suggest that perceptual distortion is correlated to certain spatial (and temporal) properties of an image (video sequence) [17, 18]. These studies lead to various techniques, called *perceptual quantization* or *adaptive perceptual quantization*, that take into account properties of the Human Visual System (HVS) in determining the quantization scale [19, 20, 21, 22, 23, 24, 25].

Based upon this body of work, we propose a separation of the quantization scale Q_{s} into a *nominal quantization* Q and a *perceptual quantization function* $P(I, Q)$ such that $Q_{\mathrm{s}} = P(I, Q)$, where I denotes the block being quantized. The function P is chosen so that if the same nominal quantization Q were used to code two blocks then the blocks would have the same perceptual distortion. In this way, the nominal quantization parameter Q would correspond directly to the perceived distortion and can serve as the object for optimization. We favor a multiplicative model where $P(I, Q) = \alpha_{\mathrm{I}} Q$. (The MPEG-2 Test Model 5 [26] also uses a multiplicative formulation while an additive formulation is proposed in [27].) Where quantization noise is less noticeable, such as in highly-textured regions, we can use a larger value for α_{I} than regions where quantization noise is more noticeable, such as in relatively uniform areas. In this regards, α_{I} can be viewed as a perceptual weighting factor. Our bit rate allocation, however, works with any monotonic perceptual quantization function.

The problem of determining $P(I, Q)$ has been studied elsewhere [19, 28] and is an active research area. It is not considered further in this paper. Here, we address the assignment of Q to each picture to give constant or near-constant quality among pictures while satisfying rate constraints imposed by the channel and decoder. We propose to compute Q at the picture level; that is, we compute one Q for each picture to be coded. Besides decreasing the computation over computing a different Q for each macroblock, this method results in constant perceptual quality within each picture. The framework can certainly be generalized to other coding units, and in principle can be applied to code other types of data, such as images and speech.

3.2 Bit-Production Modeling

For simplicity, we assume that each picture has a bit-production model that relates the picture's nominal quantization Q to the number of coded bits B. This assumes that the coding of one picture is independent of any other. This independence holds for an MPEG encoding that uses only intraframe (I) pictures, but not for one that uses forward predictive (P) or bidirectionally predictive (B) pictures, for example. In practice, the extent of the dependency is limited to small groups of pictures. Nonetheless, we initially assume independence to ease analysis and defer treatment of dependencies until a later section where we consider practical implementations.

We specify Q and B to be non-negative real-valued variables. In practice, the quantization scale Q_s and B are positive integers with $Q_s = \lfloor P(I,Q) \rfloor$. However, to facilitate analysis, we assume that there is a continuous function for each picture that maps Q to B.

For a sequence of N pictures, we define N corresponding bit-production models $\{f_1, f_2, \ldots, f_N\}$ that map nominal quantization scale to bits: $b_i = f_i(q_i)$, where $f_i : [0,\infty] \mapsto [l_i, u_i]$, with $0 \le l_i < u_i$. (We number pictures in encoding order and not temporal display order.) We require the models to have the following properties:

1. $f_i(0) = u_i$,
2. $f_i(\infty) = l_i$,
3. f_i is continuous and monotonically decreasing.

From these conditions, it follows that f_i is invertible with $q_i = g_i(b_i)$, where $g_i = f_i^{-1}$ and

$g_i : [l_i, u_i] \mapsto [0, \infty]$. We note that g_i is also continuous and monotonically decreasing. Although monotonicity does not always hold in practice, it is a generally accepted assumption.

In video coding systems, the number of bits produced for a picture also depends upon a myriad of coding choices besides quantization scale, such as motion compensation and the mode used to code each block. We assume that these choices are made independently of quantization and prior to performing rate control.

3.3 Buffering Constraints

The MPEG standards specify that an encoder should produce a bitstream that can be decoded by a hypothetical decoder referred to as the Video Buffering Verifier (VBV). With MPEG-2, data can be transferred to the VBV either at a constant or variable bit rate; whereas the MPEG-1 standard only defines VBV operation with a constant bit rate. In either mode of operation, the number of bits produced by each picture must be controlled so as to satisfy constraints imposed by the operation of the decoder buffer, whose size B_{V} is specified in the bitstream by the encoder. The encoder also specifies the maximum transfer rate R into the VBV buffer and the amount of time the decoder should wait before decoding the first picture. In this section, we consider constraints on the number of bits produced for each picture that follow from analysis of the VBV. The reader is referred to [29] for a more general discussion of buffer constraints in video coder systems.

3.3.1 Constant Bit Rate

We first examine the mode of operation in which the compressed bitstream is to be delivered at a constant bit rate R.

Definition 1 Given a sequence of N pictures, an *allocation* $s = \langle s_1, s_2, \ldots, s_N \rangle$ is an N-tuple containing bit allocations for all N pictures, so that s_n is the number of bits allocated to picture n.

Let B_{V} be the size of the decoder buffer; $B_{\mathrm{f}}(s, n)$ the fullness of the VBV buffer, resulting from allocation s, just *before* the nth picture is removed from the buffer; R the rate at which bits enter the decoding buffer; T_n the amount of time required to display picture n; and $B_{\mathrm{a}}(n) = RT_n$ the number of bits that enter the buffer in the time it takes to display picture n. For constant bit rate (CBR) operation, the state of the VBV buffer is described by the recurrence

$$\begin{aligned} B_{\mathrm{f}}(s,1) &= B_1, \\ B_{\mathrm{f}}(s,n+1) &= B_{\mathrm{f}}(s,n) + B_{\mathrm{a}}(n) - s_n, \end{aligned} \tag{1}$$

where B_1 is the initial buffer fullness. Unwinding the recurrence, we can also express (1) as

$$B_{\mathrm{f}}(s,n+1) = B_1 + \sum_{j=1}^{n} B_{\mathrm{a}}(j) - \sum_{j=1}^{n} s_j. \tag{2}$$

To prevent the decoder buffer from overflowing we must have

$$B_{\mathrm{f}}(s,n+1) \leq B_{\mathrm{V}}. \tag{3}$$

The MPEG standards allow pictures to be skipped in certain applications. We assume that all pictures are coded, in which case all bits in the encoding of picture n must arrive at the decoder by the time it is to be decoded and displayed; that is, we must have

$$B_{\mathrm{f}}(s,n) \geq s_n. \tag{4}$$

A violation of this condition is called a buffer *underflow*.

We now have an upper bound and can derive a lower bound for the number of bits that we can use to code picture n. From (1), (3), and the non-negativity of s_n, we have

$$s_n \geq \max\{B_{\mathrm{f}}(s,n) + B_{\mathrm{a}}(n) - B_{\mathrm{V}}, 0\}. \tag{5}$$

In summary, for constant bit rate operation, in order to pass video buffer verification, an allocation s must satisfy the following for all n:

$$\max\{B_{\mathrm{f}}(s,n) + B_{\mathrm{a}}(n) - B_{\mathrm{V}}, 0\} \leq s_n \leq B_{\mathrm{f}}(s,n). \tag{6}$$

An exemplary plot of the evolution of the buffer fullness over time for CBR operation is shown in Figure 1(a). In this example, the decoder waits T_0 seconds before decoding the first picture, at which time the buffer fullness is B_1. The time to display each picture is assumed to be a constant T seconds. In the plot, the upper and lower bounds for the number of bits to code picture 2 are shown as U_2 and L_2, respectively.

3.3.2 Variable Bit Rate

We now examine the scenario where the compressed video bitstream is to be delivered at a variable bit rate (VBR). Specifically, we adopt the MPEG-2 VBV model where bits always enter the decoder buffer at the peak rate R until the buffer is full. Depending upon the state of the buffer, bits enter during each display interval at a rate that is *effectively* variable up to the peak rate R. The maximum number of bits entering the buffer in the time it takes to display picture n is $B_{\mathrm{a}}(n) = RT_n$.

For VBR operation, the state of the VBV buffer is described by:

$$\begin{aligned} B_{\mathrm{f}}(s,1) &= B_{\mathrm{V}}, \\ B_{\mathrm{f}}(s,n+1) &= \min\{B_{\mathrm{V}}, B_{\mathrm{f}}(s,n) + B_{\mathrm{a}}(n) - s_n\}. \end{aligned} \tag{7}$$

Unlike the CBR case, the decoder buffer is prevented from overflowing by the minimization in (7). When $B_{\mathrm{f}}(s,n) + B_{\mathrm{a}}(n) - s_n > B_{\mathrm{V}}$, we say that picture n results in a *virtual overflow*. When a virtual overflow occurs, the effective input rate to the VBV buffer during that display interval is less than the peak rate. Like the CBR case, underflow is possible and to prevent it (4) must hold. The evolution of the buffer fullness is shown for VBR operation in Figure 1(b). The time to display each picture is assumed to be a constant T seconds. As shown in the plot, the number of bits that enter the buffer during each display interval is variable, with virtual overflows occurring for pictures 2 and 4.

MPEG-2 defines a second VBR mode wherein bits enter the buffer at a piecewise-constant rate up to the peak rate R. The rate at which bits for picture i are input to the VBV is determined by the coded **vbv_delay** parameter and the number of bits for picture i. We note that with the same bit allocation, the VBV buffer fullness for the first VBR mode is equal to or higher than for the second mode. Intuitively, if the channel rate is not further constrained, a lexicographically-optimal bit allocation for the first VBR mode should not be worse than an optimal bit allocation for the second mode, all else being equal.

3.4 Buffer-Constrained Bit-Allocation Problem

Using the bit-production model and VBV constraints defined above, we now formalize the buffer-constrained bit-allocation problem.

Definition 2 A *buffer-constrained bit-allocation problem* P is specified by a tuple

$$P = \langle N, F, B_{\mathrm{T}}, B_{\mathrm{V}}, B_1, B_{\mathrm{a}} \rangle,$$

where N is the number of pictures; $F = \langle f_1, f_2, \ldots, f_N \rangle$ is a sequence of N functions, as specified in Section 3.2, that model the relationship between the nominal quantization scale and the number of coded bits for each picture; B_{T} is the target number of bits to code all N pictures; B_{V} is the size of the VBV buffer in bits; B_1 is the number of bits initially in the VBV buffer; B_{a} is a function that gives the maximum number of bits that can enter the decoding buffer while each picture is being displayed.

Definition 3 Given a buffer-constrained bit-allocation problem $P = \langle N, F, B_{\mathrm{T}}, B_{\mathrm{V}}, B_1, B_{\mathrm{a}} \rangle$, an allocation s is a *legal allocation* if the following conditions hold:

1. $\sum_{j=1}^{N} s_j = B_{\mathrm{T}}$
2. Equation (4) holds: $B_{\mathrm{f}}(s, n) \geq s_n$.
3. For CBR only, (5) holds: $s_n \geq \max\{B_{\mathrm{f}}(s, n) + B_{\mathrm{a}}(n) - B_{\mathrm{V}}, 0\}$.

For convenience, we shall use the shorter term "bit-allocation problem" to refer to the buffer-constrained bit-allocation problem and assume that bit-allocation problems are given so that a legal allocation exists.

3.5 Lexicographic Optimality

We now formally define the lexicographic optimality criterion. As mentioned in Section 3.1, we equate nominal quantization scale with perceptual distortion and define the optimality criterion based upon the nominal quantization Q assigned to each picture.

Let S be the set of all legal allocations for a bit-allocation problem P. For an allocation $s \in S$, let $\mathbf{Q}^s = \langle Q_1^s, Q_2^s, \ldots, Q_N^s \rangle$ be the values of Q to achieve the bit allocation specified by s. Thus $Q_i^s = g_i(s_i)$, where g_i is as defined in Section 3.2. Ideally, we would like an optimal allocation to use a constant nominal quantization scale. However, this may not be feasible because of buffer constraints. We could consider minimizing an l_k norm of $\mathbf{Q}^s$. However, as discussed earlier, such an approach does not guarantee constant quality where possible and may result in some pictures having extreme values of Q_i.

Instead, we would like to minimize the maximum Q_i. Additionally, given that the maximum Q_i is minimized, we want the second largest Q_i to be as small as possible, and so on. This is referred to as *lexicographic optimality* or *lexicographic minimax* in the literature (e.g., [30]).

We define a sorted permutation DEC on $\mathbf{Q}^s$ such that for $\mathrm{DEC}(\mathbf{Q}^s) = \langle q_{j_1}, q_{j_2}, \ldots, q_{j_N} \rangle$, we have $q_{j_1} \geq q_{j_2} \geq \cdots \geq q_{j_N}$. Let $\mathrm{rank}(s,k)$ be the kth element of $\mathrm{DEC}(\mathbf{Q}^s)$; that is, $\mathrm{rank}(s,k) = q_{j_k}$. We define a binary relation $\succ$ on allocations as follows: $s = \langle s_1, \ldots, s_N \rangle \succ s' = \langle s'_1, \ldots, s'_N \rangle$ if and only if $\mathrm{rank}(s,j) = \mathrm{rank}(s',j)$ for $j = 1, 2, \ldots, k-1$ and $\mathrm{rank}(s,k) > \mathrm{rank}(s',k)$ for some $1 \leq k \leq N$. We also define $s \prec s'$ if and only if $s' \succ s$; $s \asymp s'$ if and only if $\mathrm{rank}(s,j) = \mathrm{rank}(s',j)$ for all j; $s \succeq s'$ if and only if $s \succ s'$ or $s \asymp s'$; and $s \preceq s'$ if and only if $s \prec s'$ or $s \asymp s'$.

Definition 4 A legal allocation s^* is *lexicographically optimal* if $s^* \preceq s$ for all other legal allocation s.

Lemma 1 *Given a bit-allocation problem $P = \langle N, F, B_{\mathrm{T}}, B_{\mathrm{V}}, B_1, B_{\mathrm{a}} \rangle$, if there exists a legal allocation s and a quantization q such that $g_n(s_n)$ is the constant quantization q for all n, where g_n is defined as in Section 3.2, then s is the only lexicographically-optimal allocation for P.*

Proof: First we prove that s is optimal. Since s is a legal allocation, we have $\sum_{j=1}^{N} s_j =$

$\sum_{j=1}^{N} f_j(q) = B_{\mathrm{T}}$. Suppose that s is not optimal. Let s' be an optimal allocation. Then $\mathrm{rank}(s', k) < \mathrm{rank}(s, k) = q$ for some k, and $\mathrm{rank}(s', j) \leq \mathrm{rank}(s, j)$ for all j. Therefore $s'_l > f_l(q)$ for some l and $s'_j \geq f_j(q)$ for all j since f_j is a decreasing function. Thus $\sum_{j=1}^{N} s'_j > \sum_{j=1}^{N} f_j(q) = B_{\mathrm{T}}$, a contradiction. Therefore s is optimal.

Now we show that s is the only optimal allocation. Let s' be an optimal allocation. Since s and s' are both optimal, $s \preceq s'$ and $s \succeq s'$, implying $s \asymp s'$. Then $\mathrm{rank}(s, j) = \mathrm{rank}(s', j)$ for all j. Therefore $\mathrm{rank}(s', j) = q$ for all j. Thus $s' = s$. $\square$

Lemma 1 establishes a desirable property of the lexicographic optimality criterion: If a constant-Q allocation is legal, it is the only lexicographically-optimal allocation. This meets our objective of obtaining a constant-quality allocation (via perceptual quantization) when feasible.

4 CBR Analysis

In this section, we analyze the buffer-constrained bit-allocation problem under constant-bit-rate VBV constraints, as described in Section 3.3.1.

Before proceeding with a formal theoretical treatment, we first present some intuition for the results that follow. If we consider a video sequence as being composed of segments of differing coding difficulty, a segment of "easy" pictures can be coded at a higher quality (lower distortion) than an immediately following segment of "hard" pictures if we code each segment at a constant bit rate. Since we have a decoder buffer, we can vary the bit rate to some degree, depending upon the size of the buffer. If we could somehow "move" bits from the easy segment to the hard segment, we would be able to code the easy segment at a lower quality than before and the hard segment at a higher quality, thereby reducing the difference in quality between the two segments. In terms of the decoder buffer, this corresponds to filling up the buffer during the coding of the easy pictures, which are coded with less than the average bit rate. By use of the accumulated bits in the buffer, the hard pictures can be coded with effectively more than the average bit rate.

Similarly, suppose we have a hard segment followed by an easy segment. We would like to empty the buffer during the coding of the hard pictures to use as many bits as the buffer allows to code the hard pictures at above the average bit rate. This simultaneously leaves room in the buffer to accumulate excess bits resulting from coding the easy pictures below the average bit rate.

This behavior of emptying and filling the buffer is intuitively desirable since this means that we are taking advantage of the full capacity of the buffer. In the following analysis, we will show that such a behavior is indeed exhibited by a lexicographically-optimal bit allocation.

4.1 Analysis

First, we establish a set of necessary conditions for optimality with the following lemma.

Lemma 2 *Given a CBR bit-allocation problem* $P = \langle N, F, B_{\mathrm{T}}, B_{\mathrm{V}}, B_1, B_{\mathrm{a}} \rangle$*, if* s *is an optimal allocation, the following are true:*

1. *If* $Q_j^s > Q_{j+1}^s$ *for some* $1 \le j < N$ *then* $B_{\mathrm{f}}(s, j) = s_j$.
2. *If* $Q_j^s < Q_{j+1}^s$ *for some* $1 \le j < N$ *then* $B_{\mathrm{f}}(s, j+1) = B_{\mathrm{V}}$.

Proof: A sketch of the proof of Lemma 2 is shown in Figure 2. The proof is by contradiction. In the figure, the VBV buffer is shown for a hypothetical situation in which $Q_2^s > Q_3^s$ and $Q_1^s < Q_2^s$ and the switching conditions are not met.

In the first case, for $Q_2^s > Q_3^s$, if the buffer is not empty after picture 2 is decoded, an alternate allocation can be constructed that is the same as the allocation shown except that the VBV plot follows the dotted line for the segment between pictures 2 and 3. The dotted line results from decreasing Q_2^s and increasing Q_3^s while still maintaining $Q_2^s > Q_3^s$ and not causing the buffer to underflow. This results in a better allocation than before, a contradiction. Intuitively, this corresponds to shifting bits right-to-left from a relatively easy picture (lower Q) to a relatively hard picture (higher Q). This shifting of bits can take place until the buffer becomes empty or until $Q_1^s = Q_2^s$.

In the second case, for $Q_1^s < Q_2^s$, if the buffer is not full before picture 2 is decoded, an alternate allocation can be constructed that is the same as the allocation shown except that the

VBV plot follows the dashed line for the segment between pictures 1 and 2. The dashed line results from increasing Q_1^s and decreasing Q_2^s while still maintaining $Q_1^s < Q_2^s$ and not causing the buffer to overflow. This results in a better allocation than before, a contradiction. Intuitively, this corresponds to shifting bits left-to-right from a relatively easy picture (lower Q) to a relatively hard picture (higher Q). This shifting of bits can take place until the buffer becomes full or until $Q_1^s = Q_2^s$.

□

Lemma 2 gives us a set of necessary "switching" conditions for optimality. It states that an optimal allocation consists of segments of constant Q, with changes in Q occurring only at buffer boundaries. Also, Q must change in a specific manner depending upon whether the buffer is full or empty. We observe that in an optimal allocation, the decoder buffer is full before decoding starts on a relatively difficult scene, which is marked by an increase in Q (Case 2). This policy makes the entire capacity of the decoder buffer available to code the more difficult pictures. On the other hand, before decoding a relatively easy scene, which is marked by a decrease in Q (Case 1), the buffer is emptied in order to provide the most space to accumulate bits when the easy scene uses less than the average bit rate. These observations agree with the intuitions provided earlier. We also note that Lemma 1 follows directly from Lemma 2.

The theorem that follows is the main result of this section and shows that the switching conditions are also sufficient for optimality.

Theorem 1 *Given a CBR bit-allocation problem $P = \langle N, F, B_{\mathrm{T}}, B_{\mathrm{V}}, B_1, B_{\mathrm{a}} \rangle$, a legal allocation s is optimal if and only if the following conditions hold. Also, the optimal allocation is unique.*

1. *If $Q_j^s > Q_{j+1}^s$ for some $1 \leq j < N$, then $B_{\mathrm{f}}(s, j) = s_j$.*

2. *If $Q_j^s < Q_{j+1}^s$ for some $1 \leq j < N$, then $B_{\mathrm{f}}(s, j+1) = B_{\mathrm{V}}$.*

Proof: Lemma 2 established these condition as necessary for optimality. Now we need to show that these conditions are also sufficient and imply uniqueness.

Let s be a legal allocation that meets both conditions of the theorem. Let $Q^s_{(k)}$ denote the kth highest value of Q assigned by allocation s. We first consider maximal segments of pictures that are allocated the maximum Q, $Q^s_{(1)}$. From Lemma 2, we have that each such segment starts with the buffer either full or at the initial state and ends with the buffer either empty or at the final state. Since these segments start at the maximum buffer level, we cannot use a lower value of Q for any picture in such a segment without causing a buffer underflow. Therefore an optimal allocation must use $Q^s_{(1)}$ for these segments of pictures.

We can proceed by induction on k to prove that segments of pictures assigned $Q^s_{(k)}$ by allocation s are given the same Q in an optimal allocation. We conclude that s has the same allocation as any optimal allocation, and therefore the optimal allocation is unique. □

Detailed proofs of Lemma 2 and Theorem 1 can be found in [31].

4.2 Related Work

Conditions similar to the switching conditions of Theorem 1 have been described in [32] for optimal buffered bit allocation under a minimum sum-distortion criterion and assuming independent convex rate-distortion functions. In this work, the Lagrange multiplier method is used to find a bit allocation that is optimal within a convex-hull approximation. The optimal vector of Lagrange multipliers consists of constant-valued segments that increase (decrease, respectively) only when the decoder buffer is full (empty, respectively).

In [33, 34], the theory of *majorization* [35] is applied to reduce the variability in transmission rate for stored video. In this setting, the problem is to determine a feasible transmission schedule by which a pre-compressed video bitstream can be transmitted over a communications channel to the decoder without underflowing or overflowing the decoder buffer. As applied to this problem, majorization results in minimizing the peak and variance in transmission rate. It can be easily shown that majorization is in fact equivalent to lexicographic minimization of the transmission schedule, subject to the constraint that the total number of bits transmitted is fixed.

5 CBR Allocation Algorithms

Theorem 1 is a powerful result. It says that to find the optimal allocation we need only to find a legal allocation that meets the stated switching conditions. In Section 5.1, we first use the technique of dynamic programming (DP) to develop a simple algorithm to compute a lexicographically-optimal CBR allocation in polynomial time and linear space. The DP algorithm uses a "brute-force" approach that does not take advantage of some of the structure inherent in the problem. In Section 5.2, we develop an efficient algorithm that exploits the monotonicity of the bit-production models to compute the optimal bit allocation in linear time.

5.1 DP Algorithm

The basic idea behind dynamic programming is to decompose a given problem in terms of optimal solutions to smaller problems. All we need to do is maintain invariant the conditions stated in Theorem 1 for each subproblem we solve. We do this by constructing optimal bit allocations for pictures 1 to k that end up with the VBV buffer in one of two states: *full* or *empty*. These states are exactly the states where a change in Q may occur. Let Top^k be the optimal allocation for pictures 1 to k that end up with the VBV buffer *full*, if such an allocation exists. Similarly, let Bot^k be the optimal allocation for pictures 1 to k that end up with the VBV buffer *empty*. Suppose that we have computed Top^i and Bot^i for $1 \le i \le k$. To compute Top^{k+1}, we search for a legal allocation among $\left\{\emptyset, \mathrm{Top}^1, \ldots, \mathrm{Top}^k, \mathrm{Bot}^1, \ldots, \mathrm{Bot}^k\right\}$, where $\emptyset$ denotes the empty allocation, to which we can concatenate a constant-Q segment to give a legal allocation s such that the switching conditions are met and the buffer ends up full, that is, $B_{\mathrm{f}}(s, k+1) = B_{\mathrm{V}}$. Similarly, for Bot^{k+1} we search for a previously computed allocation that, when extended by a constant-Q segment, meets the switching conditions and results in the buffer being empty, that is, $B_{\mathrm{f}}(s, k+1) = s_{k+1}$.

The basic step in the DP algorithm is illustrated in Figure 3. The round nodes represent buffer states for which we have previously computed optimal allocations. Each node stores the last Q used in the optimal allocation for that state and the origin of the last constant-Q segment leading to that state. The square node represents the next state that we wish to compute. The dashed lines represent a constant-Q allocation that connects the respective nodes. To compute a solution for the square node, we need to search for an edge that connects the square node with a round node such that the switching conditions are met. For each edge, the switching conditions are checked by comparing the Q used for the edge against the last Q used in the optimal solution for the round node that the edge connects. The allocation implied by each edge is also checked for VBV compliance.

Once we have computed Top^{N-1} and Bot^{N-1}, we can compute the optimal allocation for all N pictures in a process similar to the one above for computing Top^{k} and Bot^{k}, except that the final allocation results in a final buffer state that gives the desired target number of bits B_{T}.

5.1.1 Correctness of DP Algorithm

When computing Top^{k} and Bot^{k} for $1 \leq k \leq N-1$, we have insured that the conditions of Theorem 1 are met. Additionally in the final computation, the conditions are also met. The result is a legal allocation that meets the conditions of Theorem 1 and is thus optimal.

5.1.2 Constant-Q Segments

We have used the concept of a constant-Q segment extensively in the above discussion. We now formalize this concept. First, we define a family of bit-production functions $\{F_{i,j}(q)\}$ that gives the number of bits resulting from allocating a constant value of Q for pictures i to j, inclusive:

$$F_{i,j}(q) = \sum_{i \leq k \leq j} f_k(q). \tag{8}$$

What we are really interested in, though, is the inverse of $F_{i,j}$. We denote the inverse as $G_{i,j}$ so that $G_{i,j} = F_{i,j}^{-1}$. Then $G_{i,j}(B)$ gives the constant Q that results in B bits being produced by pictures i to j collectively. Since f_i is monotonically decreasing, so is $F_{i,j}$, and thus $G_{i,j}$ is monotonically increasing.

5.1.3 Verifying a Constant-Q Allocation

The DP algorithm for CBR bit allocation needs to verify whether a constant-Q allocation meets VBV buffer constraints. This can be done in time linear in the length of the allocation by simulating the VBV. In the DP algorithm, $O(N^2)$ verifications of constant-Q allocations are needed. If each verification requires linear time, this translates to at least cubic time complexity for the DP algorithm.

We observe that the constant-Q allocations to be verified start with the buffer either full, empty, or at its initial state; and end with the buffer either full, empty, or at its final state. We also note that for an allocation to a segment of pictures, say from i to j, with a fixed initial buffer state, say B_1, and using B_{T} bits, there is a continuous range of Q values that results in a legal allocation. When additional pictures are considered, this range of legal Q values never widens. Furthermore, the upper bound for Q is simply the minimum Q among the constant-Q allocations for pictures i to j in which the buffer is exactly full for some picture k, where $i \le k < j$. More formally,

$$G_{i,j}(B_{\mathrm{T}}) \le \min_{i \le k < j} G_{i,k}\Big(B_1 + \sum_{i \le m \le k} B_{\mathrm{a}}(m) - B_{\mathrm{V}}\Big). \tag{9}$$

Similarly, the lower bound for Q is the maximum Q among the constant-Q allocations for pictures i to j in which the buffer is exactly empty for some picture k, where $i \le k < j$. More formally,

$$G_{i,j}(B_{\mathrm{T}}) \ge \max_{i \le k < j} G_{i,k}\Big(B_1 + \sum_{i \le m < k} B_{\mathrm{a}}(m)\Big). \tag{10}$$

We can use these observations to perform all the VBV verifications in constant time per verification with linear-time preprocessing.

5.1.4 Time and Space Complexity

The time complexity of the DP algorithm depends upon two main factors: the time to compute a constant-Q allocation and the time to verify whether a sub-allocation is legal.

We assume that f_i and $G_{i,j}$ can be evaluated in constant time with $O(N)$ preprocessing time and space. An example is $f_i(q) = \alpha_i/q + \beta_i$, where

$$G_{i,j}(B) = \frac{\sum_{i \le k \le j} \alpha_k}{B - \sum_{i \le k \le j} \beta_k}.$$

We can precompute the prefix sums of α_i and β_i in linear time and space and then use these to compute $G_{i,j}$ in constant time. The same technique can be used for bit-production models of the form: $f_i(q) = \alpha_i/q^2 + \beta_i/q + \gamma_i$, $f_i(q) = \alpha_i/q^3 + \beta_i/q^2 + \gamma_i/q + \phi_i$, and $f_i(q) = \alpha_i/q^4 + \beta_i/q^3 + \gamma_i/q^2 + \phi_i/q + \theta_i$. Examples of other functional forms for f_i with a closed-form solution for $G_{i,j}$ can be found in [36]. Of course, we need to insure that the models are monotonically decreasing.

Since VBV verification and constant-Q calculation can be done in constant time with linear-time preprocessing, computing Top^k and Bot^k takes $O(k)$ time. Therefore, to compute an optimal allocation for a sequence of N pictures would take $\sum_{k=1}^{N} O(k) = O(N^2)$ time. If we store pointers for tracing the optimal sequence of concatenations, the algorithm requires $O(N)$ space.

5.2 Linear-Time CBR Algorithm

The dynamic programming solution in Section 5 ignores some of the structure that exists in the framework and focuses solely on achieving the switching conditions of Theorem 1 by "brute force." For example, a "blind" search strategy is used to find a feasible constant-Q segment that connects the end state with a previously-computed optimal sub-allocation and meets the switching conditions.

A linear-time and linear-space algorithm for optimal smoothing of transmission rates has been described in [34] and is based on an algorithm for computing the shortest path in the presence of rectilinear barriers [37]. By plotting transmitted bits versus time, we can view a transmission schedule as a curve in 2D space. Buffering constraints are manifested as rectilinear barriers. As shown in [33], an optimally smooth transmission schedule follows the shortest path that does not cross the barriers.

A lexicographically-optimal bit allocation does not have such a simple geometric interpretation since the relationship between the nominal quantization scale and rate is non-linear. However, optimal rate smoothing and optimal bit allocation do have some properties in common. Conditions analogous to the switching conditions of Theorem 1 exist in the rate smoothing context [34]: the decoder buffer must be full when there is an increase in the transmission rate and empty when there is a decrease.

To apply the linear-time algorithm to lexicographic bit allocation, we equate a constant-bit-rate (constant-slope) segment in the rate-smoothing problem with a constant-Q segment in the bit-allocation problem. An increase (decrease) in rate would correspond to an increase (decrease) in Q.

6 VBR Analysis

In this section, we analyze the buffer-constrained bit-allocation problem under variable-bit-rate VBV constraints, as described in Section 3.3.2. The analysis leads to an efficient iterative algorithm for computing a lexicographically-optimal solution.

In CBR operation, the total number of bits that a CBR stream can use is dictated by the channel bit rate and the buffer size. With VBR operation, the total number of bits has no lower bound, and its upper bound is determined by the peak bit rate and the buffer size. Consequently, VBR is useful and most advantageous over CBR when the average bit rate needs to be lower than the peak bit rate. This is especially critical in storage applications, where the storage capacity, and not the transfer rate, is the limiting factor. Another important application of VBR video coding is for multiplexing multiple video bitstreams over a CBR channel [38]. In this application, statistical properties of the multiple video sequences may allow more VBR bitstreams with a given peak rate R to be multiplexed onto the channel than CBR bitstreams coded at a constant rate of R.

For typical VBR applications, then, the average bit rate is lower than the peak. In this case, bits enter the decoder buffer at an effective bit rate that is less than the peak during the display interval of many pictures. In interesting cases, there will be segments of pictures that are coded with an average bit rate higher than the peak. This is possible because of the buffering. During the display of these pictures, the VBV buffer fills at the peak rate. Since these pictures require more bits to code than the peak rate, they are "harder" to code than the other "easier" pictures.

In order to equalize quality, the easy pictures should be coded at the same base quality. It does not pay to code any of the hard pictures at a quality higher than that of the easy pictures. The bits expended to do so could instead be better distributed to raise the quality of the easy pictures. Among the hard pictures, there are different levels of coding difficulty. Using the same intuitions from the CBR case, we can draw similar conclusions about the buffer emptying and filling behavior among the hard pictures.

In the following analysis, we show that a lexicographically-optimal VBR bit allocation possesses the properties described above. In particular, the hard segments of pictures in a VBR bit allocation behave as in a CBR setting. In fact, the VBR algorithm invokes the CBR algorithm to allocate bits to segments of hard pictures.

6.1 Analysis

The following two lemmas characterize the "easy" pictures in an optimal allocation, that is, the pictures that are coded with the best quality (lowest Q).

Lemma 3 *Given a VBR bit-allocation problem* $P = \langle N, F, B_{\mathrm{T}}, B_{\mathrm{V}}, B_{\mathrm{V}}, B_{\mathrm{a}} \rangle$ *and an optimal allocation* s*, if* $B_{\mathrm{f}}(s,j) + B_{\mathrm{a}}(j) - s_j > B_{\mathrm{V}}$ *for* $1 \le j \le N$*, then* $Q_j^s = \min_{1 \le k \le N} Q_k^s$.

The above lemma states that, in an optimal allocation, pictures that cause a virtual overflow (see Section 3.3.2) are coded with the globally minimum Q, Q_{min}^s. We can prove the lemma by contradiction. For a picture i that causes a virtual overflow and is not coded with Q_{min}^s, we can shift bits between picture i and another picture j that is coded with Q_{min}^s so that the quality of picture i will be improved but not beyond the quality of picture j. Such a shift in bits will result in a lexicographically better allocation.

As discussed above, there may be segments of hard pictures that require coding at higher than the peak rate R. The following lemma gives a set of *switching* conditions for changes in Q that are similar to the results of Lemma 2 and characterize the behavior of hard pictures.

Lemma 4 *Given a VBR bit-allocation problem* $P = \langle N, F, B_{\mathrm{T}}, B_{\mathrm{V}}, B_{\mathrm{V}}, B_{\mathrm{a}} \rangle$, *if* s *is an optimal allocation, the following are true:*

1. *If* $Q_j^s > Q_{j+1}^s$ *for* $1 \le j < N$, *then* $B_{\mathrm{f}}(s,j) = s_j$.
2. *If* $Q_j^s < Q_{j+1}^s$ *for* $1 \le j < N$, *then* $B_{\mathrm{f}}(s,j+1) = B_{\mathrm{V}}$ *and* $B_{\mathrm{f}}(s,j+1) + B_{\mathrm{a}}(j+1) - s_{j+1} \le B_{\mathrm{V}}$.

The proof for Lemma 4 is similar to that for Lemma 2 except that for Case 2, we use Lemma 3 to show that $B_{\mathrm{f}}(s^*, j+1) + B_{\mathrm{a}}(j+1) - s^*_{j+1} \le B_{\mathrm{V}}$.

The following theorem is the main result of this section. It shows that the minimum-Q and switching conditions in the previous lemmas are also sufficient for optimality.

Theorem 2 *Given a VBR bit-allocation problem* $P = \langle N, F, B_{\mathrm{T}}, B_{\mathrm{V}}, B_{\mathrm{V}}, B_{\mathrm{a}} \rangle$, *a legal allocation* s *is optimal if and only if the following conditions hold. Also, the optimal allocation is unique.*

1. *If* $B_{\mathrm{f}}(s,j) + B_{\mathrm{a}}(j) - s_j > B_{\mathrm{V}}$ *for* $1 \le j \le N$, *then* $Q_j^s = \min_{1 \le k \le N} Q_k^s$.
2. *If* $Q_j^s > Q_{j+1}^s$ *for* $1 \le j < N$, *then* $B_{\mathrm{f}}(s,j) = s_j$.
3. *If* $Q_j^s < Q_{j+1}^s$ *for* $1 \le j < N$, *then* $B_{\mathrm{f}}(s,j+1) = B_{\mathrm{V}}$ *and* $B_{\mathrm{f}}(s,j+1) + B_{\mathrm{a}}(j+1) - s_{j+1} \le B_{\mathrm{V}}$.

Lemmas 3 and 4 establish these as necessary conditions. The proof for sufficiency and uniqueness is similar to that of Theorem 1 except for segments with the minimum Q, Q^s_{min}. Using the same arguments as in the proof of Theorem 1, we can show that hard pictures k that are assigned $Q_k^s > Q^s_{\mathrm{min}}$ use the same number of bits as in an optimal allocation. This leaves the same number of bits to be allocated by s to the remaining easy pictures as in the optimal allocation. Since the bit-production models are monotonic, there is exactly one such assignment using a constant Q.

Although Theorem 2 is an important result, it does not show us how to compute the minimum Q with which to code the "easy" pictures. The following lemmas and theorem show that, if we relax the bit budget constraint, we can find the minimum Q, and therefore the optimal allocation, to meet the bit budget by an iterative process. Furthermore, the iterative process is guaranteed to converge to the optimal allocation in a finite number of steps.

Lemma 5 *Given two VBR bit-allocation problems* $P^{(1)} = \langle N, F, B_{\mathrm{T}}^{(1)}, B_{\mathrm{V}}, B_{\mathrm{V}}, B_{\mathrm{a}} \rangle$ *and* $P^{(2)} = \langle N, F, B_{\mathrm{T}}^{(2)}, B_{\mathrm{V}}, B_{\mathrm{V}}, B_{\mathrm{a}} \rangle$ *that have optimal allocations* $s^{(1)}$ *and* $s^{(2)}$*, respectively, with* $B_{\mathrm{T}}^{(1)} < B_{\mathrm{T}}^{(2)}$*, then* $s^{(1)} \succ s^{(2)}$*.*

This lemma states the intuitively obvious: If we use more bits we expect to get a lexicographically better allocation. The following lemma is more interesting and states that pictures that are already hard retain their bit allocation with an increase in the bit budget.

Lemma 6 *Given two VBR bit-allocation problems* $P^{(1)} = \langle N, F, B_{\mathrm{T}}^{(1)}, B_{\mathrm{V}}, B_{\mathrm{V}}, B_{\mathrm{a}} \rangle$ *and* $P^{(2)} = \langle N, F, B_{\mathrm{T}}^{(2)}, B_{\mathrm{V}}, B_{\mathrm{V}}, B_{\mathrm{a}} \rangle$ *that have optimal allocations* $s^{(1)}$ *and* $s^{(2)}$*, respectively, with* $B_{\mathrm{T}}^{(1)} < B_{\mathrm{T}}^{(2)}$*, then* $s_j^{(1)} = s_j^{(2)}$ *for* j *such that* $Q_j^{s^{(1)}} > \min_{1 \le k \le N} Q_k^{s^{(1)}}$*.*

The hard pictures are constrained by the buffer and the peak rate and not by the bit budget. Intuitively, increasing the bit budget, therefore, cannot improve the allocation to these pictures. We can prove this lemma using the same techniques to prove Theorem 1.

Lemma 7 *Given two VBR bit-allocation problems* $P^{(1)} = \langle N, F, B_{\mathrm{T}}^{(1)}, B_{\mathrm{V}}, B_{\mathrm{V}}, B_{\mathrm{a}} \rangle$ *and* $P^{(2)} = \langle N, F, B_{\mathrm{T}}^{(2)}, B_{\mathrm{V}}, B_{\mathrm{V}}, B_{\mathrm{a}} \rangle$ *that have optimal allocations* $s^{(1)}$ *and* $s^{(2)}$*, respectively, with* $B_{\mathrm{T}}^{(1)} < B_{\mathrm{T}}^{(2)}$*, then* $\min_{1 \le k \le N} Q_k^{s^{(1)}} > \min_{1 \le k \le N} Q_k^{s^{(2)}}$*.*

This lemma follows from Lemma 5 and 6. We summarize Lemmas 5, 6, and 7 with the following theorem.

Theorem 3 *Given two VBR bit-allocation problems* $P^{(1)} = \langle N, F, B_{\mathrm{T}}^{(1)}, B_{\mathrm{V}}, B_{\mathrm{V}}, B_{\mathrm{a}} \rangle$ *and* $P^{(2)} = \langle N, F, B_{\mathrm{T}}^{(2)}, B_{\mathrm{V}}, B_{\mathrm{V}}, B_{\mathrm{a}} \rangle$ *that have optimal allocations* $s^{(1)}$ *and* $s^{(2)}$*, respectively, with* $B_{\mathrm{T}}^{(1)} < B_{\mathrm{T}}^{(2)}$*, then*

1. $s^{(1)} \succ s^{(2)}$,
2. $s_j^{(1)} = s_j^{(2)}$ *for* j *such that* $Q_j^{s^{(1)}} > \min_{1 \le k \le N} Q_k^{s^{(1)}}$*, and*
3. $\min_{1 \le k \le N} Q_k^{s^{(1)}} > \min_{1 \le k \le N} Q_k^{s^{(2)}}$.

7 VBR Allocation Algorithm

Theorems 2 and 3 give us a way to find the optimal allocation for a given VBR allocation problem. If we know the minimum Q that the optimal allocation uses, then it would be straightforward to find the optimal allocation. However, in general we do not know what that minimum Q would be. Theorem 3 gives us an iterative way to find the minimum Q.

7.1 VBR Algorithm

Here we sketch an iterative algorithm for computing a VBR allocation.

1. Mark all pictures as *easy*. Let $B_{\text{easy}} \leftarrow B_{\text{T}}$.

2. Allocate B_{easy} bits to easy pictures using a constant Q. Let $Q_{\min}$ be the value of Q used.

3. Simulate operation of VBV to identify *hard* and *easy* segments of pictures. A hard segment contains pictures that lead to a buffer underflow and consists of pictures that follow the most recent virtual overflow up to and including the picture that caused the overflow. In case a virtual overflow has not yet occurred, the segment starts with the first picture. After identifying a hard segment, reset the buffer fullness to empty and continue the simulation.

4. Assign a bit budget to each newly identified hard segment such that the underflow is just prevented. By preventing underflow in the hard segments, we are left with extra unallocated bits.

5. Let B_{hard} be the total number of bits allocated to hard pictures. Let $B_{\text{easy}} \leftarrow B_{\text{T}} - B_{\text{hard}}$.

6. If a new hard segment has been identified in Step 3, goto Step 2.

7. Allocate bits to maximal segments of hard pictures using the CBR algorithm.

7.2 Correctness of VBR Algorithm

Here we prove that the VBR algorithm computes a lexicographically-optimal allocation. We do this by showing that the algorithm computes an allocation that satisfies the switching conditions of Theorem 2.

First, we make several observations about the VBR algorithm.

1. Pictures marked "easy" are assigned the same value of Q,

2. "Hard" pictures are marked in segments that start either at the beginning of the video sequence or with the buffer full and that end with the buffer empty.

3. Segments of hard pictures are allocated using the CBR algorithm.

The correctness of the CBR algorithm insures that within hard segments Conditions 2 and 3 of Theorem 2 hold. In order to show that Condition 1 also holds, we first need to show that the CBR algorithm does not assign a Q lower than the Q_{min} computed in Step 2.

Lemma 8 *Let s be the allocation computed by the VBR algorithm. Let i and j denote the indices of the beginning and end, respectively, of a hard segment as identified in Step 3. Then* $\min_{i \le k \le j} Q_k^s \ge Q_{\mathrm{min}}$.

Proof: Let s' be an allocation that is the same as s except for pictures i to j, where s' uses Q_{min}. Thus, in a VBV simulation using s' for pictures i to j, s' does not cause a virtual overflow and underflows only at picture j. Let u and v mark the beginning and end, respectively, of a segment with the minimum Q in the CBR allocation for pictures i to j. We consider two cases for u: $u = i$ and $u > i$. If $u = i$, then we have $B_{\mathrm{f}}(s, u) = B_{\mathrm{f}}(s', u)$ since $s_k = s'_k$ for $k < i$. If $u > i$, then since u marks the beginning of a segment with minimum Q in the CBR allocation for pictures i to j, from Theorem 1, $B_{\mathrm{f}}(s, u-1) = s_{u-1}$. This implies that $B_{\mathrm{f}}(s, u) = B_{\mathrm{a}}(u-1)$. Since s' does not cause an underflow for picture $u-1$, $B_{\mathrm{f}}(s', u-1) \ge s'_{u-1}$, which implies that $B_{\mathrm{f}}(s', u) \ge B_{\mathrm{a}}(u-1)$. In either case, we have

$$B_{\mathrm{f}}(s', u) \ge B_{\mathrm{f}}(s, u). \tag{11}$$

We consider two cases for v: $v = j$ and $v < j$. If $v = j$, then $B_{\mathrm{f}}(s', v) < s'_v$ since an underflow occurs at picture j. Thus $B_{\mathrm{f}}(s', v+1) < B_{\mathrm{a}}(v)$. But since s is a legal allocation, $B_{\mathrm{f}}(s, v+1) \ge B_{\mathrm{a}}(v)$. If $v < j$, then since v marks the end of a segment with minimum Q in the CBR allocation for pictures i to j, from Theorem 1, $B_{\mathrm{f}}(s, v+1) = B_{\mathrm{V}}$. Since s' does not cause virtual overflow, $B_{\mathrm{f}}(s', v+1) \leq B_{\mathrm{V}}$. In either case,

$$B_{\mathrm{f}}(s', v+1) \leq B_{\mathrm{f}}(s, v+1). \tag{12}$$

Through some algebraic manipulations we have

$$\sum_{k=u}^{v} s_k = \sum_{k=u}^{v} B_{\mathrm{a}}(k) + B_{\mathrm{f}}(s, u) - B_{\mathrm{f}}(s, v+1), \tag{13}$$

$$\sum_{k=u}^{v} s'_k = \sum_{k=u}^{v} B_{\mathrm{a}}(k) + B_{\mathrm{f}}(s', u) - B_{\mathrm{f}}(s', v+1). \tag{14}$$

Combining (11), (12), (13), and (14) we have

$$\sum_{k=u}^{v} s_k \leq \sum_{k=u}^{v} s'_k. \tag{15}$$

Pictures u to v use a constant Q in both allocations s and s', where s uses $Q = \min_{i \leq k \leq j} Q_k^s$ and s' uses $Q_{\min}$. Therefore we have

$$F_{u,v}\left(\min_{i \leq k \leq j} Q_k^s\right) \leq F_{u,v}(Q_{\min}). \tag{16}$$

Since $F_{u,v}$ is a monotonically decreasing function (see Section 5.1.2), we have $\min_{i \leq k \leq j} Q_k^s \geq Q_{\min}$. □

From Lemma 8, we can conclude that after each iteration of the VBR algorithm, $Q_{\min}$ is indeed the minimum Q. Since hard segments do not include pictures that cause a virtual overflow, Condition 1 of Theorem 2 also holds.

Theorem 4 *Each pass through the VBR algorithm results in an allocation that is lexicographically optimal for the number of bits actually allocated.*

7.3 Time and Space Complexity

We note that the loop in the VBR algorithm terminates when no more hard segments are identified. This implies that the algorithm terminates after at most N iterations, where N is the number of pictures.

Not counting the executions of the CBR algorithm, each iteration of the VBR algorithm takes $O(N)$ time and space. Since at most $O(N)$ iterations are performed, the time complexity excluding the executions of the CBR algorithm is $O(N^2)$. Assuming that $G_{i,j}$ can be evaluated in constant time, an optimal CBR allocation can be computed in linear time and space. Therefore the time complexity of the VBR algorithm is $O(N^2)$.

When there are relatively few hard segments, computing an optimal VBR allocation will likely be faster in practice than computing a CBR allocation. Furthermore, Theorem 4 guarantees that we can halt the VBR algorithm after any number of iterations and have an optimal allocation. The decision to continue depends upon whether the achieved bit consumption is acceptable. With each iteration the number of bits allocated increases.

8 Implementation

In this section, we describe an implementation of rate control using the lexicographically-optimal bit-allocation algorithms presented in Sections 5 and 7 within the ISO MPEG-2 software encoder [39]. With this implementation, we aim to: 1) verify the effectiveness of lexicographic optimality, 2) assess the practical implications of the assumptions made in the framework, namely independent coding and continuous variables, 3) explore various bit-production models, and 4) develop robust techniques for recovering from errors due to the approximate models and the simplifying assumptions.

8.1 Perceptual Quantization

For perceptual quantization, we use the TM5 adaptive quantization scheme, where the nominal quantization scale is modulated by an *activity factor* that is computed from the spatial variance of the luminance blocks within a macroblock. In TM5, the actual quantization scale used for coding a particular macroblock is determined from an initially computed (global) reference quantization scale, a (local) feedback factor that is dependent of the state of a virtual encoding buffer, and the normalized activity factor for that block. For modeling purposes, we define the nominal quantization Q for a picture as the average of the product of the reference quantization scale and the buffer-feedback factor over all coded macroblocks.

8.2 Bit-Production Modeling

The framework in Section 3 presumes the existence of an exact continuous bit-production model for each picture. In practice, the rate-distortion function of a complex encoding system, such as MPEG, cannot be determined exactly for non-trivial classes of input. Therefore, approximate models are used in practice.

As the complexity analyses in Sections 5.1.4 and 7.3 show, the running time for the optimal bit-allocation algorithms depends on the time to evaluate $G_{i,j}$, the function that is used to compute a constant-Q sub-allocation. In practice, therefore, the chosen models should admit efficient computation of $G_{i,j}$.

8.2.1 Hyperbolic Model

In [19], the following "hyperbolic" model forms the basis of an adaptive bit-allocation algorithm:

$$f_i(q_i) = \frac{\alpha_i}{q_i} + \beta_i, \tag{17}$$

where α_i is associated with the complexity of coding picture i and β_i with the overhead for coding the picture. The hyperbolic model is one of the simplest models to exhibit the monotonicity and concavity characteristic of rate-distortion functions. TM5 adopts a similar model where only the complexity term is used. With adaptive quantization techniques, α_i and β_i are typically estimated from the results of encoding previous pictures. The parameters can also be determined by coding a sampling of blocks in picture i and fitting the parameters to the coding statistics. There is a simple closed-form expression for $G_{i,j}$:

$$G_{i,j}(b) = \frac{\sum_{i \le k \le j} \alpha_k}{b - \sum_{i \le k \le j} \beta_k}. \tag{18}$$

As previously discussed in Section 5.1.4, we can precompute the cumulative sums for α_i and β_i in linear time and space and then use these to compute $G_{i,j}$ in constant time.

In related work, Ding and Liu [40] propose the following more general class of bit-production models and describe its use in rate control:

$$f_i(q) = \frac{\alpha_i}{q^{\gamma_i}} + \beta_i. \tag{19}$$

The extra parameter γ_i is dependent on the picture type (I, P, or B) and is intended to capture the different rate-distortion characteristics for each picture type. One drawback to (19) is that the model is non-linear with respect to the parameters, and we know of no closed-form solution to $G_{i,j}$ in the general setting. Although numerical techniques can be used to solve for $G_{i,j}$, this could adversely affect the computational efficiency of the bit-allocation algorithms.

8.2.2 Linear-Spline Model

In preliminary experiments, we have found that the hyperbolic model works well with small changes in the quantization scale Q between pictures. However, with a large variation in Q between successive pictures, as may occur with a scene change, the model becomes less reliable. This is because the model is defined by only two parameters α_i and β_i. We can compensate for this by performing multiple encoding passes to ensure that the parameters are determined close to the actual operating point. We now consider a different approach where more effort is expended initially to construct more accurate bit models that are then used to encode the video sequence in a single pass.

Lin, Ortega, and Kuo [41] have proposed using cubic-spline interpolation models of rate and distortion in conjunction with a gradient-based rate-control algorithm [42]. The spline models are computed by first encoding each picture several times using a select set of M quantization scales, $\{x_1,\ x_2,\ \ldots,\ x_M\}$ with $x_1 < x_2 < \cdots < x_M$, and measuring the actual rates. Each quantization/rate pair is called a *control point*. For picture i, the bit-production model between two consecutive control points (x_k, y_k) and (x_{k+1}, y_{k+1}) is approximated as

$$f_i^k(x_k) = a_{ik}x^3 + b_{ik}x^2 + c_{ik}x + d_{ik}. \tag{20}$$

The authors suggest using the Fibonacci-like set {1, 2, 3, 5, 8, 13, 21, 31} for the control quantization scales to exploit the exponential-decay typical of rate-distortion functions.

One drawback of a cubic-spline model is that it is generally not monotonic. To ensure monotonicity, we consider a simpler linear-spline interpolation model, where a line segment is used to interpolate the bit-production function between control points. For picture i, the function between two consecutive control points (x_k, y_k) and (x_{k+1}, y_{k+1}) has the form

$$f_i^k(x) = \alpha_{i,k}x + \beta_{i,k}. \tag{21}$$

In case the control points themselves do not exhibit monotonicity, we enforce monotonicity by skipping those control points that violate the monotonicity property. For quantization scales less than x_1 or greater than x_M, we extrapolate using the parameters $(\alpha_{i,1}, \beta_{i,1})$ or $(\alpha_{i,M-1}, \beta_{i,M-1})$, respectively.

The linear-spline model has a simple closed-form expression for $G_{i,j}$ if we know the two control points that bracket the operating point. Because of the monotonicity property, we can determine the two bracketing points using binary search. Between the control points x_k and x_{k+1}, $G_{i,j}$ can be computed as

$$G_{i,j}(b) = \frac{b - \sum_{i \le m \le j} \beta_{m,k}}{\sum_{i \le m \le j} \alpha_{m,k}}. \tag{22}$$

If $x_k \le G_{i,j} \le x_{k+1}$ then the correct operating point has been found. If $G_{i,j} < x_k$, the operating point must lie between two control points with lower indices. Similarly, if $G_{i,j} > x_{k+1}$, the operating point must lie between two control points with higher indices. Since there are a fixed number of control points, we can compute $G_{i,j}$ in constant time with linear-time preprocessing.

8.2.3 Piecewise-Hyperbolic Model

In earlier experiments [43, 31], we found that the linear-spline model gave consistently better results than the simple hyperbolic model. This is because the linear-spline model has more parameters and can better approximate a picture's rate-distortion characteristics over a wider range of quantization. The hyperbolic model, on the other hand, better matches the rate-distortion locally. These two observations suggest that we can construct a more accurate model by combining the two. We can do this by replacing the linear interpolation in the spline model with hyperbolic interpolation. Instead of (21), we use

$$f_i^k(x) = \frac{\alpha_{i,k}}{x} + \beta_{i,k}. \tag{23}$$

We can compute $G_{i,j}$ using (18).

In Section 9, we present results of encoding simulations using the piecewise-hyperbolic model.

8.3 Picture-Level Rate Control

Even with accurate bit-production models, the actual number of bits produced will inevitably depart from the model, especially if we use predicted P- and B-pictures. There are essentially two ways to cope with bit-modeling errors.

8.3.1 Closed-Loop Rate Control

A popular approach taken in TM5 is to regulate the quantization scale at the macroblock level while coding a picture so that the desired bit allocation is met. This is achieved with a *closed-loop* feedback mechanism using the fullness of a virtual encoder buffer to control the macroblock quantization. One drawback of this technique is that the coded quality within a picture may vary considerably, especially for a picture that contains regions of varying complexity. With gross errors in the bit-production models, the actual average quantization scale may differ markedly from the desired quantization scale, thereby adversely affecting the coded quality.

8.3.2 Open-Loop Rate Control

Another approach is to perform *open-loop* control where the assigned (nominal) quantization scale is used to code a picture. We can then adjust the bit allocation of the remaining uncoded pictures to compensate for the difference between desired and actual bit production. An advantage of this approach is that the quality is more constant within a picture. In addition, less processing is required to code each picture. A disadvantage is that, since the bit production is not controlled below the picture layer, the actual bit production may vary from the target. If left uncorrected, the errors can accumulate and potentially cause the buffer to overflow or underflow.

After coding a picture, we can recover from errors in the bit-production model by reallocating bits to the remaining pictures optimally (for the given models). If we use the DP algorithm, instead of recomputing an optimal allocation from scratch and incurring an extra factor of N in the time complexity, we can take advantage of dynamic programming to increase the time complexity by only a constant factor. We do this for a CBR allocation and for hard pictures in a VBR allocation by constructing the dynamic programming table in the CBR algorithm *in reverse*.

As presented in Section 5, the dynamic programming algorithm works by solving for sub-allocations for pictures 1 to k for increasing values of k. We can also rework the dynamic programming to compute optimal sub-allocations for pictures k to N for decreasing values of k. We do this by computing optimal allocations that start with the buffer empty or full at picture k and ends with the buffer at the final buffer state after picture N.

We can compute a revised allocation for picture k, after encoding picture $k-1$, by searching for a proper constant-Q connector starting with the known VBV buffer fullness before picture k is removed. With the reverse dynamic programming table available, this search consumes $O(N)$ time. The total additional time to recover from bit-production errors is then $O(N^2)$, the same as the time complexity for computing the initial allocation with the DP algorithm.

As an alternative, we could use the linear-time CBR algorithm and recompute the allocation after coding each picture. This would also require $O(N^2)$ time.

The above procedure applies to a CBR allocation and to hard pictures in a VBR allocation (which are allocated using the CBR algorithm). For easy pictures in a VBR allocation, we can simply recompute a new value for $Q_{\min}$. Here, we assume that errors in bit-production modeling are not severe enough to change the classification of hard and easy pictures.

8.3.3 Hybrid Rate Control

In early experiments, we observed that closed-loop rate control resulted in rapid fluctuations in the nominal quantization scale between pictures owing to the buffer-feedback mechanism. With accurate bit-production models, however, the need to perform low-level rate control below the picture level is questionable. This suggests using open-loop control. Since we use independent bit-production models, we can expect more errors in the models at predicted pictures where the assigned Q changes. With these observations, we propose a hybrid rate control strategy where closed-loop control is used for pictures at the boundaries of a constant-Q segment and open-loop control is used for the rest. Another motivation for using closed-loop control for boundary pictures is that the VBV buffer should be either nearly empty or nearly full for these pictures, and the bit rate must be carefully controlled to avoid underflowing or overflowing the buffer.

8.4 Buffer Guard Zones

Even with the picture-level rate-control strategies outlined above, there is still the possibility of the VBV buffer overflowing or underflowing. To safeguard against this, we compute a bit allocation using a slightly smaller buffer than that specified in the MPEG bitstream so that we can have *guard zones* near the top and bottom of the buffer. For the experiments with CBR, have we chosen to place the guard zones at 5% and 95% of maximum buffer size. For VBR mode, the upper guard zone is not needed since buffer overflow is not a concern.

8.5 Limiting Lookahead

The above rate-control algorithms compute an allocation for the entire video sequence. This may not be feasible when the sequence consists of many pictures, as in a feature-length movie, for example. One way to deal with this is to partition the sequence into blocks consisting of a small number of consecutive pictures. Optimal allocation can then be performed on the blocks separately. In order to do this, the starting and ending buffer fullness must be specified for each block for the CBR case. For the VBR case, the bit budget must also be specified for each block. This approach is globally suboptimal; however, it is easy to parallelize since the block allocations are independent of each other.

Another approach is to use limited lookahead in conjunction with hybrid rate control. Using a lookahead window of size W and a step size $S \leq W$, the procedure is as follows:

1. Compute a bit allocation for the next W pictures not yet coded.
2. Code the next S pictures using hybrid rate control.
3. Repeat Step 1.

This procedure can be thought of as performing lookahead with a sliding window.

Another approach similar to the hybrid rate-control method is to use the allocation computed from a given model and only recompute the allocation when the buffer fullness reach preset buffer boundaries, such as 10% and 90% of buffer fullness. As with hybrid rate control, reverse dynamic programming can be used to speed up the reallocation.

8.6 Related Work

Several heuristic methods have been proposed in [6] to reduce the complexity as compared to an optimal bit allocation based on the Viterbi algorithm. A Lagrangian optimization technique is applied to recompute an allocation incrementally for each picture, similar to the technique described in Section 8.3.2. In addition, the Lagrangian optimization is performed with a finite window size. In essence, this method implements limited lookahead with a sliding window, similar to the technique described in Section 8.5. The authors also describe the heuristic of recomputing a allocation only when the buffer reaches predefined threshold levels.

In [10], a bit-production model is derived for block-transform coders based on rate-distortion theory and assuming a stationary Gaussian process. The model is applied for VBR coding with motion JPEG and H.261 coders. In [44], an adaptive tree-structured piecewise linear bit-production model is proposed and applied to MPEG video coding using a one-pass encoding strategy. A cubic-spline model of rate and distortion is proposed in [41] for use with a gradient-based rate-control algorithm [42] that attempts to minimize MSE. The model takes into account the temporal dependencies introduced by predictive coding.

9 Encoding Simulations

To assess the behavior and effectiveness of the lexicographic bit-allocation algorithms, the bit-production models, and the rate control strategies outlined above, we conducted encoding simulations using a two-minute (3,660 frames) promotional video clip courtesy of IBM Corporation. The interlaced video is sampled spatially at 720×480 pixels and temporally at 29.97 frames/sec (59.94 fields/sec). The clip starts with a fade-in to a spokeswoman standing in front of a slowing changing background. A block diagram in one corner of the picture then rotates and zooms to fill the screen. The diagram then remains static with some illumination changes before fading back to the spokeswoman. On one side of the picture, a collage of different video clips scroll up the screen. One of the clips then zooms to occupy the full picture. The clips cycle through a variety of action-filled scenes from horses running to a skydiver rotating on a skateboard to a bicycle race and finally to highlights from a basketball game.

We implemented the lexicographic rate-control algorithms within the software encoder provided by the MPEG-2 Simulation Group [39]. The piecewise-hyperbolic model of Section 8.2.3 is used in conjunction with the non-linear quantizer scale of MPEG-2. Since the non-linear quantizer scale already performs an "exponential-type" mapping, we use the following set of MPEG-2 **quantizer_scale_codes** $\{1, 4, 9, 13, 18, 22, 27, 31\}$ which corresponds to the set of quantization scales {1, 4, 10, 18, 32, 48, 80, 112}. We define nominal quantization based on **quantizer_scale_code** instead of on the actual quantization scale because the former is the parameter coded in the MPEG-2 bitstream.

As a reference, we also ran the simulations with TM5 rate control. A minor modification was made to the TM5 model in that the levels of the virtual encoding buffers used to regulate the quantization scale are restricted to the range $[0, 2r]$, where r is the reaction parameter given by $r = 2 \cdot$ **bit_rate/picture_rate**.

9.1 Independent Coding

To assess the performance of the lexicographic bit allocation algorithms, we initially performed encoding simulations using only I-pictures to maintain independence. For CBR mode, we specified an average bit rate of 7.0 Mbits/sec. For VBR, we used the same average rate and a peak rate of 9.0 Mbits/sec. The VBV buffer size was set to 1,835,008 bits.

The results of the encodings are presented in Table 1 and Figures 4 to 6. The table collects some summary statistics for the various coders. Figure 4 shows the evolution of the buffer fullness, Figure 5 plots the computed and observed nominal quantization, and Figure 6 plots the PSNR.

As evident from Figure 4, the TM5 coder uses only a fraction of the VBV buffer and maintains the buffer relatively level. In contrast, the lexicographic coders make better use of the VBV buffer.

The lexicographic CBR coder is able to control the quantization to a narrower range than the TM5 coder, with a resulting increase in PSNR. The lexicographic VBR coder sacrifices quality in earlier pictures in order to code better the later more complex pictures. The result is that the nominal quantization is nearly constant and the PSNR plot is more even.

Visually, the lexicographic VBR coder produced near-constant-quality video with few noticeable coding artifacts. In contrast, both CBR coders produced noticeable blocking artifacts in scenes with high motion, especially in the basketball scene. However, the lexicographic CBR coder fared noticeably better than TM5 at maintaining constant quality through scene changes and reducing artifacts during complex scenes of short duration.

9.2 Dependent Coding

In a second experiment, we performed the encoding simulations using I-, P-, and B-pictures. In order to reduce factors that would affect the actual bit production, full-search motion estimation was initially performed using a fixed nominal quantization of 13, and the same motion vectors were then used for all the encodings. The coding decisions, however, were still determined on-line. To partially compensate for the difference in the rate-distortion characteristics of the dependent P- and B-pictures, the K_{P} and K_{B} factors of TM5 are used in the perceptual quantization. An open GOP structure of length 15 with at most two consecutive B-pictures is used.

We coded the sequence in CBR mode at 3.0 Mbits/sec and in VBR mode at 3.0 Mbits/sec average and 4.5 Mbits/sec peak. The VBV buffer size is set to 1,835,008 bits. We used piecewise-hyperbolic models in conjunction with the hybrid rate-control strategy.

A summary of some encoding statistics are listed in Table 2. The buffer fullness, nominal Q, and PSNR are plotted in Figures 7, 8, and 9, respectively. The results are similar to the independent coding simulations. It is noteworthy that the difference between the initially computed lexicographic nominal quantizers and the actual values used is barely noticeable. This suggests that the independent models can be used successfully, albeit suboptimally, with dependent coding. Since we can view coding dependencies as contributing to errors in the (independent) bit-production models, the error-recovery techniques of Section 8.3 effectively allow us to apply the bit-allocation framework to predictive video coders.

10 Bit Allocation with a Discrete Set of Quantizers

One of the assumptions made in Section 3 is that there is a continuous relationship between quantization (distortion) and rate. As shown in Sections 4 and 6, this assumption facilitates rigorous analysis of the buffer-constrained bit-allocation problem under the lexicographic optimality criterion and results in an elegant characterization of the optimal solution. In order to apply directly the results of the analysis, we need to construct a continuous model of the relationship between quantization and rate. As demonstrated in Section 8, this can be done by gathering statistics during multiple encoding passes and fitting these to a chosen functional form. Because of the inevitable error in the modeling, some form of error recovery is needed, such as the methods proposed in Section 8.

In most practical coders, however, both the set of available quantizers and the number of bits produced are discrete and finite. The problem of buffer-constrained bit-allocation under these conditions have been examined by Ortega, Ramchandran, and Vetterli [45]. They provide a dynamic programming algorithm to find a CBR allocation that minimizes a sum-distortion metric. In this section, we briefly describe their algorithm and show how it can be readily extended to perform lexicographic minimization.

10.1 Dynamic Programming

The dynamic programming algorithm described in [45] is based on the Viterbi Algorithm described in [1] for solving the budget-constrained bit-allocation problem. To handle the additional buffer constraints, the buffer fullness is recorded at each state instead of the total number of bits used so far; for CBR coding, the number of bits used can be determined from the buffer fullness. We can use the recurrence equations in Section 3.3.1 to update the buffer fullness and create a trellis. Instead of pruning states that exceed a given bit budget, we instead prune states that overflow or underflow the buffer. At each stage in the construction of the trellis, we compare the current sum distortion associated with edges that enter a new state and record the minimum distortion along with a pointer to the source state. At the last stage of trellis construction, we identify the state with the minimum sum distortion and backtrack through the stored pointers to recover an optimal bit allocation. Since an integral number of bits is generated, the maximum number of states that can be generated at each stage is equal to the size of the buffer. Therefore, with M quantizers, N pictures, and a buffer of size B, the dynamic programming algorithm of [45] requires $O(MBN)$ time to compute an optimal bit allocation.

10.2 Lexicographic Extension

It is straightforward to modify the dynamic programming algorithm to perform lexicographic minimization. Instead of keeping track of a minimum sum distortion value, a scalar, we keep track of a lexicographic minimum, a vector. A naive implementation would store a vector of length k for a state at the kth stage in the trellis, where the vector records the quantizers used for coding the first k pictures. However, since the set of quantizers is finite and we are only concerned with the number of times a given quantizer is used and not with the order in which the quantizers are used, we only need to store M values at each state, where M is the number of quantizers. Each of these M values count the number of times a given quantizer has been used to code the first k pictures in an optimal path ending at the given state. Given two vectors of quantizer counts, a lexicographic comparison can be performed in $O(M)$ time. With this modification, we can find a lexicographically-optimal bit allocation in $O(M^2BN)$ time.

11 Summary

In this paper, we have introduced a novel lexicographic framework for bit allocation of MPEG video. Designed explictly to equalize the quality of pictures in a video sequence, the framework consists of independent bit-production models, buffering constraints based on the MPEG Video Buffering Verifier, and a novel lexicographic optimality criterion. Rigorous analysis under the framework reveals necessary and sufficient conditions for optimality that are simple and intuitive. These conditions lead to efficient bit-allocation algorithms for both constant-bit-rate and variable-bit-rate operation.

Experimental implementations of these algorithms confirm the theoretical analysis and produce encodings that are more uniform in quality than that achieved with existing rate control methods. With the implementations, we explore several forms of bit-production models and propose a hybrid rate-control strategy that allows for robust recovery from errors in the bit-production modeling. We performed simulations of both independent and dependent coding. The results suggest that, although developed for independent coding, the framework can successfully be applied to dependent coding.

Finally, we show how an existing technique for bit allocation with a discrete set of quantizers can be extended to perform lexicographic optimization with minimal overhead.

12 Acknowledgements

This work was started by the first two authors at IBM T. J. Watson Research Center. Continued funding for the first author was provided in part by Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant F49620–94–1–0217. Support for the third author was provided in part by Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant F49620–94–1–0217 and by an associate membership in CESDIS.

References

[1] Y. Shoham and A. Gersho, Efficient bit allocation for an arbitrary set of quantizers, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 36, no. 9, pp. 1445–1453, Sept. 1988.

[2] A. J. Viterbi and J. K. Omura, *Principles of Digital Communication and Coding*, McGraw-Hill, 1979.

[3] G. D. Forney, The viterbi algorithm, *Proceedings of the IEEE*, vol. 61, pp. 268–278, Mar. 1973.

[4] K. M. Uz, J. M. Shapiro, and M. Czigler, Optimal bit allocation in the presence of quantizer feedback, in *Proceedings 1993 International Conference on Acoustics, Speech and Signal Processing*, 1993, vol. 5, pp. 385–388.

[5] G. M. Schuster and A. K. Katsaggelos, *Rate-Distortion Based Video Compression: Optimal Video Frame Compression and Object Boundary Encoding*, Kluwer Academic Publishers, Boston, MA, 1997.

[6] A. Ortega, K. Ramchandran, and M. Vetterli, Optimal buffer-constrained source quantization and fast approximations, in *Proceedings 1992 International Symposium on Circuits and Systems*, San Diego, CA, May 1992, pp. 192–195.

[7] K. Ramchandran, A. Ortega, and M. Vetterli, Bit allocation for dependent quantization with applications to MPEG video coders, in *Proceedings 1993 International Conference on Acoustics, Speech and Signal Processing*, 1993, vol. 5, pp. 381–384.

[8] C.-Y. Hsu, A. Ortega, and A. R. Reibman, Joint selection of source and channel rate for VBR video transmission under ATM policing constraints, *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 6, Aug. 1997.

[9] W. Ding, Joint encoder and channel rate control of VBR video over ATM networks, Apr. 1996, preprint.

[10] J.-J. Chen and H.-M. Hang, A transform video coder source model and its application, in *Proceedings ICIP'94*, 1994, vol. 2, pp. 967–971.

[11] J. B. Ghosh, Siting facilities along a line when equity of service is desirable, *Journal of Operation Research Society*, vol. 47, no. 3, pp. 435–445, Mar. 1996.

[12] R. S. Klein, H. Luss, and D. R. Smith, Lexicographic minimax algorithm for multiperiod resource allocation, *Mathematical Programming*, vol. 55, no. 2, pp. 213–234, June 1992.

[13] M. Luptacik and F. Turnovec, Lexicographic geometric programming, *European Journal of Operational Research*, vol. 51, no. 2, pp. 259–269, Mar. 1991.

[14] E. Marchi and J. A. Oviedo, Lexicographic optimality in the multiple objective linear programming. The nucleolar solution, *European Journal of Operational Research*, vol. 57, no. 3, pp. 355–359, Mar. 1992.

[15] W. Ogryczak, On the lexicographic minimax approach to location–allocation problems, Tech. Rep. IS - MG 94/22, Université Libre de Bruxelles, Dec. 1994.

[16] A. Premoli and W. Ukovich, Piecewise lexicographic programming. A new model for practical decision problems, *Journal of Optimization Theory and Applications*, vol. 72, no. 1, pp. 113–142, Jan. 1992.

[17] L. A. Olzak and J. P. Thomas, Seeing spatial patterns, in *Handbook of Perception and Human Performance*, K. Boff, L. Kaufman, and J. Thomas, Eds. Wiley, 1986.

[18] V. R. Algazi, Y. Kato, M. Miyahara, and K. Kotani, Comparison of image coding techniques with a picture quality scale, in *Proceedings of SPIE, Applications of Digital Image Processing XV*, San Diego, CA, July 1992, pp. 396–405.

[19] E. Viscito and C. Gonzales, A video compression algorithm with adaptive bit allocation and quantization, in *SPIE Proceedings: Visual Communications and Image Processing*, Nov. 1991, vol. 1605, pp. 58–72.

[20] A. Puri and R. Aravind, Motion-compensated video coding with adaptive perceptual quantization, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 1, no. 4, pp. 351–361, Dec. 1991.

[21] N. S. Jayant, J. Johnson, and R. Safranek, Signal compression based on models of human perception, *Proceedings of the IEEE*, vol. 81, pp. 1385–1422, Oct. 1993.

[22] T.-Y. Chung, K.-H. Jung, Y.-N. Oh, and D.-H. Shin, Quantization control for improvement of image quality compatible with MPEG2, *IEEE Transactions on Consumer Electronics*, vol. 40, no. 4, pp. 821–825, Nov. 1994.

[23] F.-H. Lin and R. M. Mersereau, An optimization of MPEG to maximize subjective quality, in *Proceedings ICIP'95*, 1995, vol. 2, pp. 547–550.

[24] S. J. P. Westen, R. L. Lagendijk, and J. Biemond, Perceptual optimization of image coding algorithms, in *Proceedings ICIP'95*, 1995, vol. 2, pp. 69–72.

[25] G. Cicalini, L Favalli, and A. Mecocci, Dynamic psychovisual bit allocation for improved quality bit rate in MPEG-2 transmission over ATM links, *Electronic Letters*, vol. 32, no. 4, pp. 370–371, Feb. 1996.

[26] ISO-IEC/JTC1/SC29/WG11/N0400, Test model 5, Apr. 1993, Document AVC-491b, Version 2.

[27] M. R. Pickering and J. F. Arnold, A perceptually efficient VBR rate control algorithm, *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 527–532, Sept. 1994.

[28] K. W. Chun, K. W. Lim, H.D. Cho, and J. B. Ra, An adaptive perceptual quantization algorithm for video coding, *IEEE Transactions on Consumer Electronics*, vol. 39, no. 3, pp. 555–558, Aug. 1993.

[29] A. R. Reibman and B. G. Haskell, Constraints on variable bit-rate video for ATM networks, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 2, no. 4, pp. 361–372, Dec. 1992.

[30] T. Ibaraki and N. Katoh, *Resource Allocation Problems*, MIT Press, Cambridge, MA, 1988.

[31] D. T. Hoang, *Fast and Efficient Algorithms for Text and Video Compression*, PhD thesis, Brown University, May 1997.

[32] D. W. Lin and J.-J. Chen, Efficient bit allocation under multiple constraints on cumulated rates for delayed video coding, in *Visual Communications and Image Processing '97*, J. Biemond and E. J. Delp, Eds., Feb. 1997, pp. 1370–1381, Proc. SPIE 3024.

[33] J. D. Salehi, Z.-L. Zhang, J. F. Kurose, and D. Towsley, Supporting stored video: Reducing rate variability and end-to-end resource requirements through optimal smoothing, in *Proceedings 1996 ACM International Conference on Measurement and Modeling of Computer Systems (ACM SIGMETRICS)*, May 1996.

[34] J. D. Salehi, *Scheduling Network Processing on Multimedia and Multiprocessor Servers*, PhD thesis, University of Massachussetts Amherst, Sept. 1996.

[35] A. W. Marshall and I. Olkin, *Inequalities: Theory of Majorization and its Applications*, Academic Press, 1979.

[36] H. Luss and S. K. Gupta, Allocation of effort resources among competitive activities, *Operations Research*, vol. 23, pp. 360–366, 1975.

[37] D. T. Lee and F. P. Preparata, Euclidean shortest paths in the presence of rectilinear barriers, *Networks*, vol. 14, pp. 393–410, 1984.

[38] B. G. Haskell and A. R. Reibman, Multiplexing of variable rate encoded streams, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 4, no. 4, pp. 417–424, Aug. 1994.

[39] MPEG Software Simulation Group, MPEG-2 encoder/decoder version 1.1a, July 4, 1994, URL: `http://www.mpeg.org/MSSG`.

[40] W. Ding and B. Liu, Rate control of MPEG video coding and recording by rate-quantization modeling, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 1, pp. 12–20, Feb. 1996.

[41] L.-J. Lin, A. Ortega, and C.-C. J. Kuo, Cubic spline approximation of rate and distortion functions for MPEG video, in *Digital Video Compression: Algorithms and Technologies 1996*,

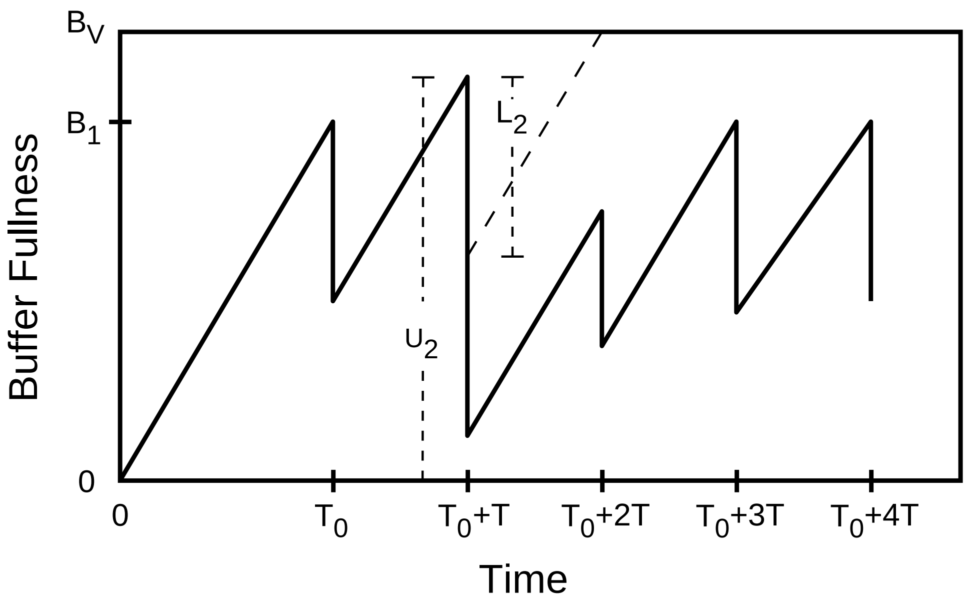
V. Bhaskaran, F. Sijstermans, and S. Panchanathan, Eds., 1996, pp. 169–180, Proc. SPIE 2668.

[42] L.-J. Lin, A. Ortega, and C.-C. J. Kuo, A gradient-based rate control algorithm with applications to MPEG video, in *Proceedings ICIP'95*, Washington, D.C., Oct. 1995.

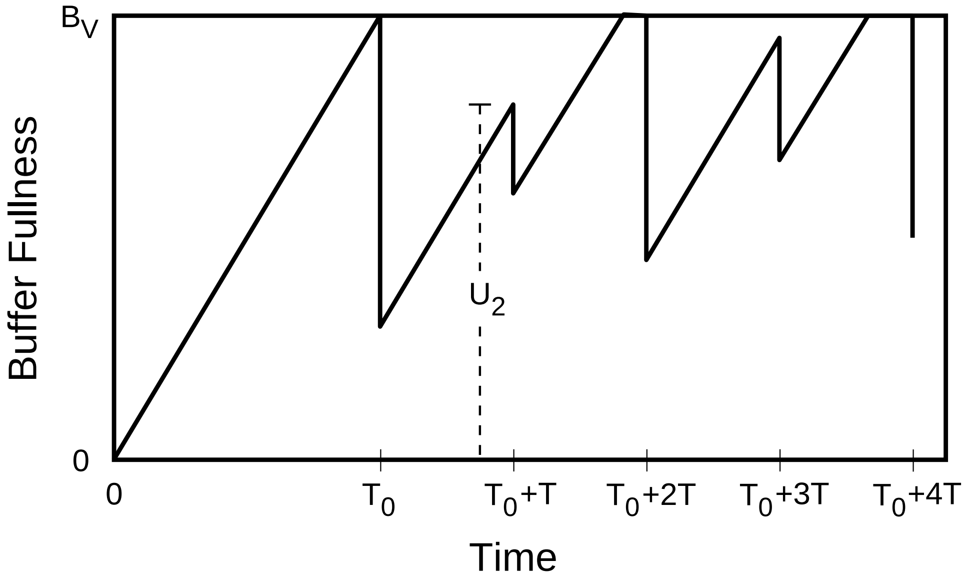
[43] D. T. Hoang, E. Linzer, and J. S. Vitter, Lexicographically optimal rate control for video coding with MPEG buffer constraints, Tech. Rep. CS–1996–02, Duke University, Dept. of Computer Science, 1996.

[44] J.-B. Cheng and H.-M. Hang, Adaptive piecewise linear bits estimation model for MPEG based video coding, in *Proceedings ICIP'95*, 1995, vol. 2, pp. 551–554.

[45] A. Ortega, K. Ramchandran, and M. Vetterli, Optimal trellis-based buffered compression and fast approximations, *IEEE Transactions on Image Processing*, vol. 3, no. 1, pp. 26–40, Jan. 1994.



(a) CBR Operation



(b) VBR Operation

Figure 1: Sample plots of buffer fullness for CBR and VBR operation.

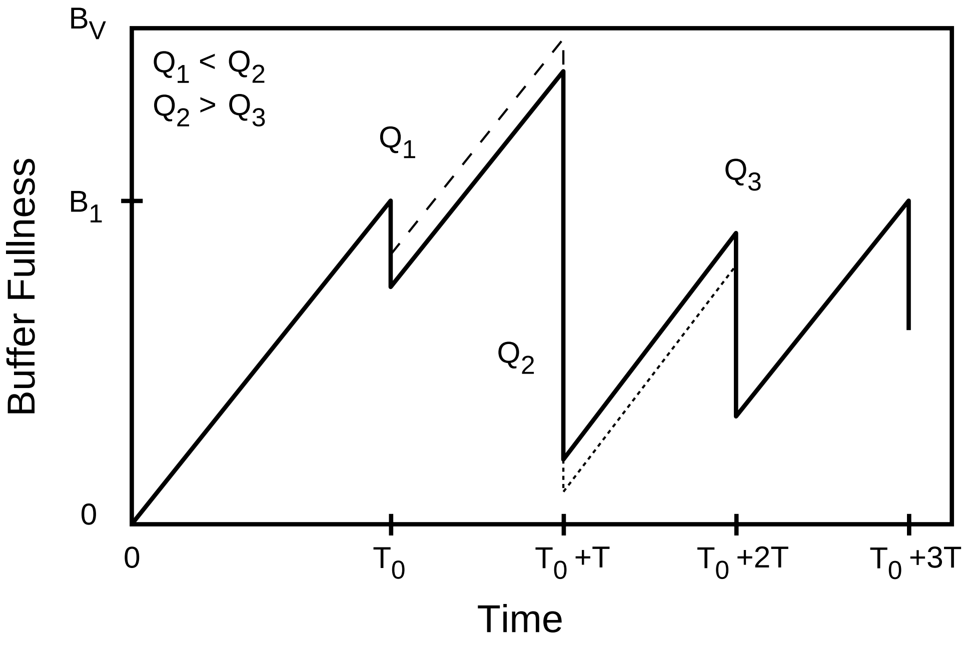


Figure 2: Sketch for proof of Lemma 2.

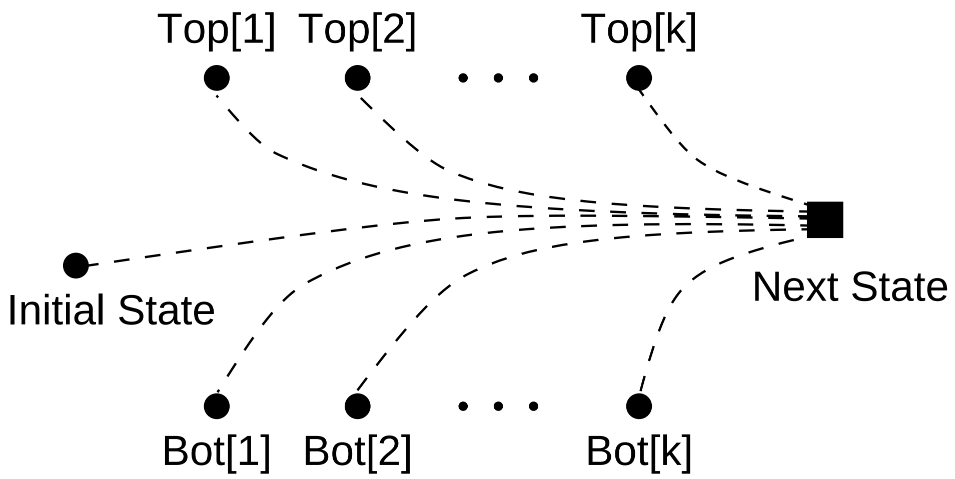
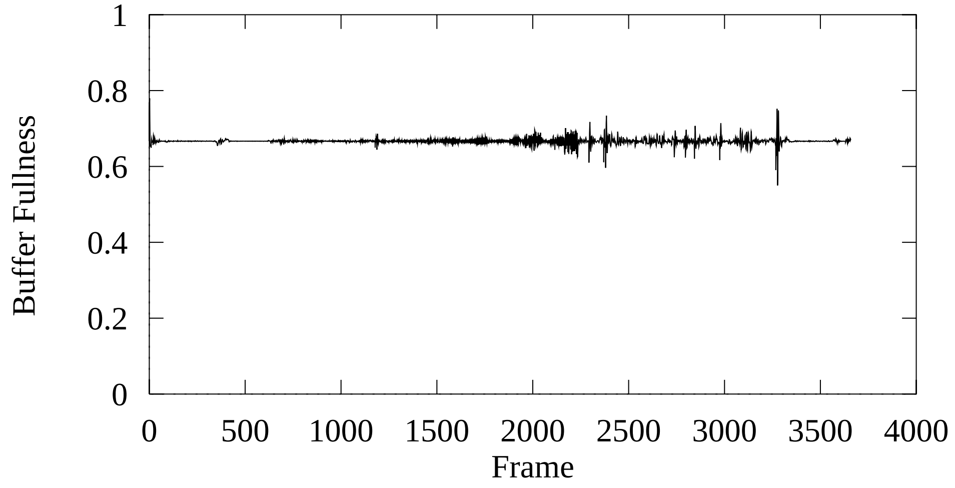


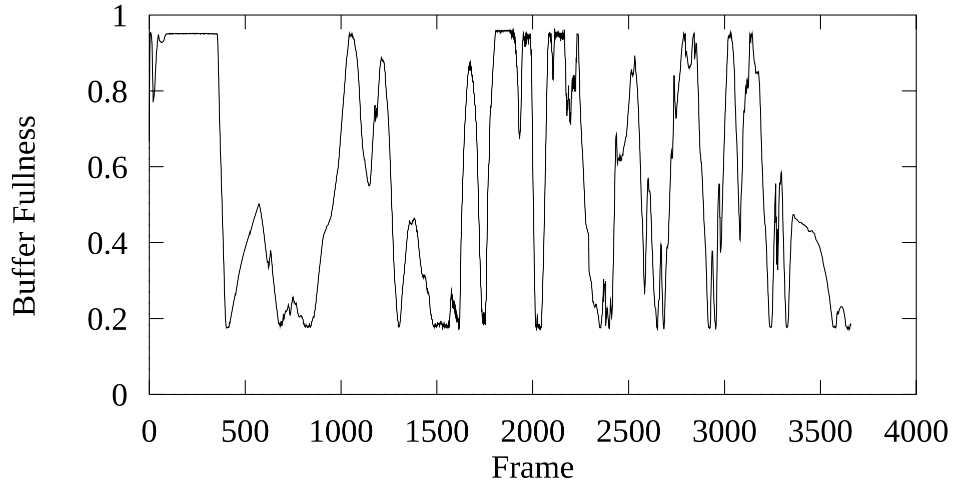
Figure 3: Illustration of search step in dynamic programming algorithm.

Table 1: Results of independent coding simulations.

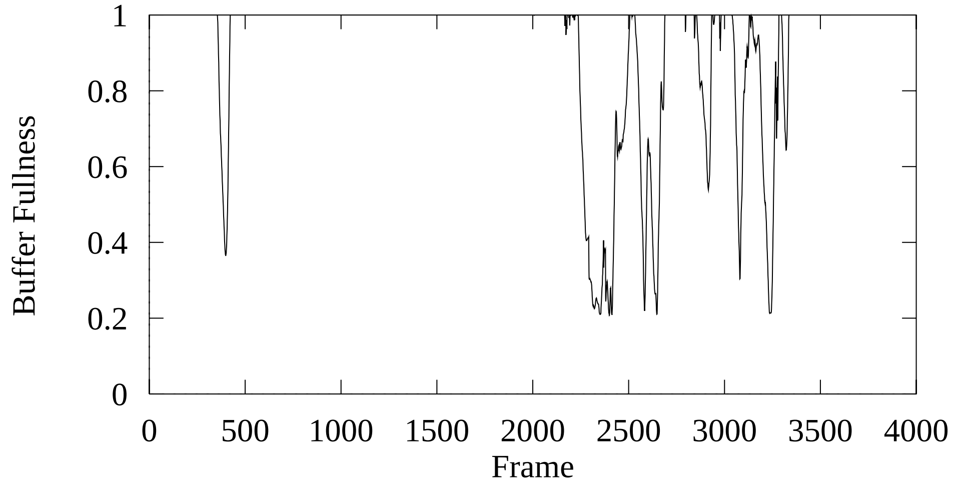
Method	Average PSNR (dB)	Std. Dev. of PSNR	Average Nom. Q	Std. Dev. of Nom. Q	Maximum Nom. Q	Minimum Nom. Q
TM5 CBR	33.53	3.35	12.79	4.91	36.38	0.48
Lexicographic CBR	33.56	3.13	12.81	4.10	23.17	0.97
Lexicographic VBR	33.52	1.99	12.65	0.94	17.02	11.76



(a) TM5 CBR

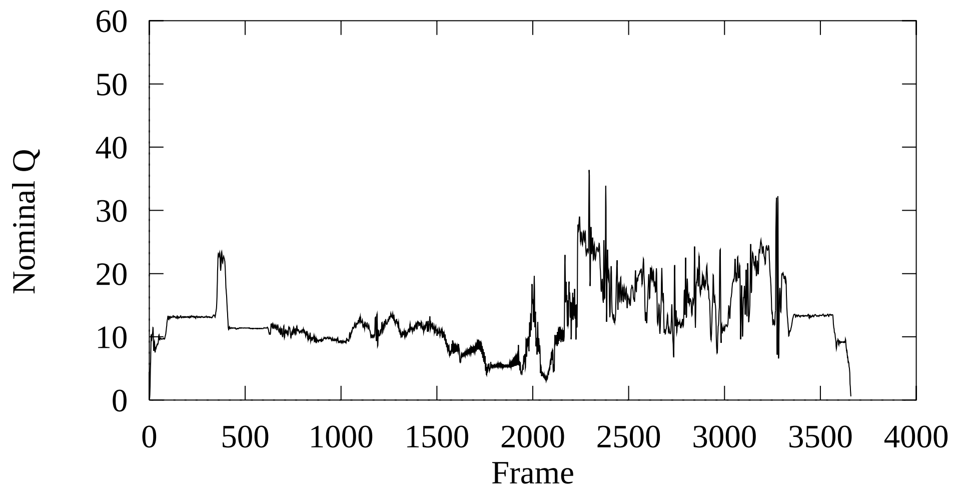


(b) Lexicographic CBR

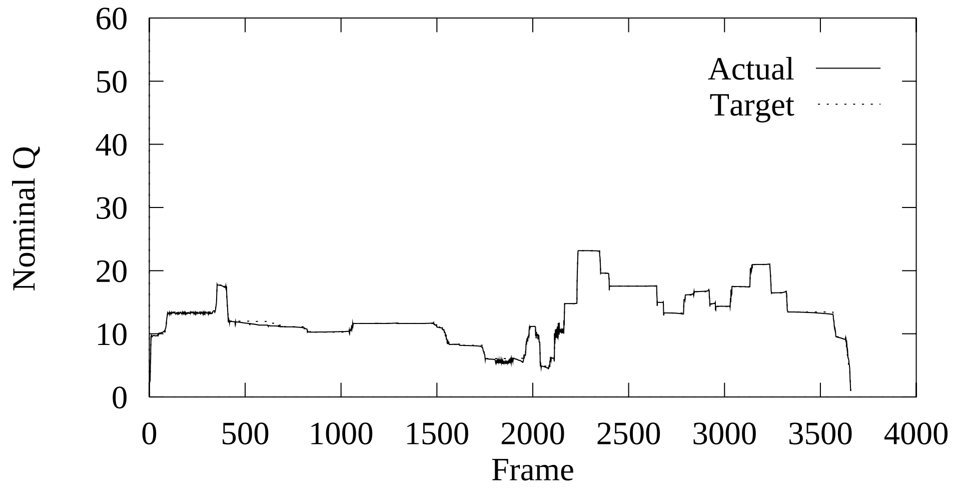


(c) Lexicographic VBR

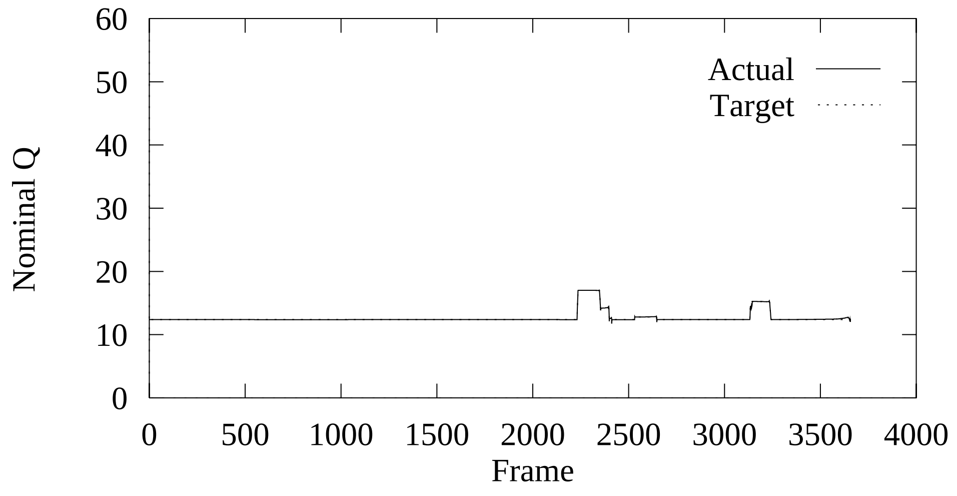
Figure 4: Evolution of normalized buffer fullness for independent coding simulations.



(a) TM5 CBR

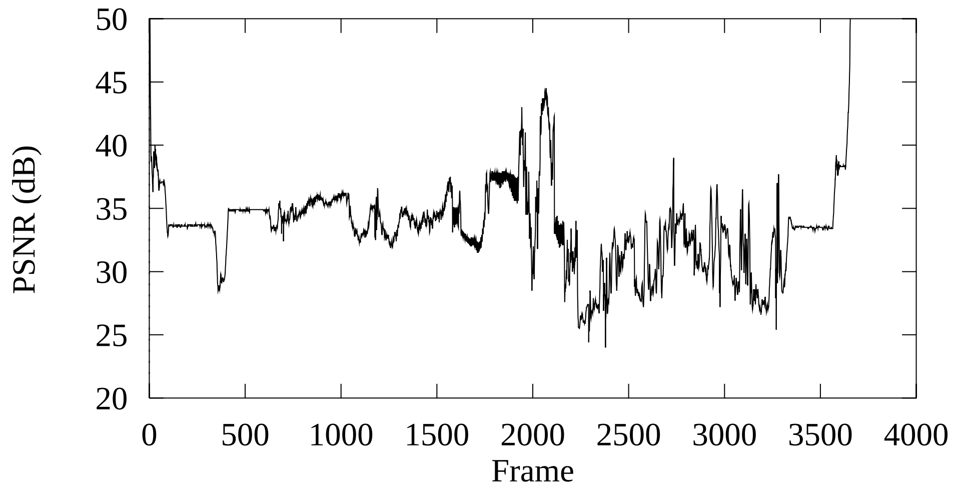


(b) Lexicographic CBR

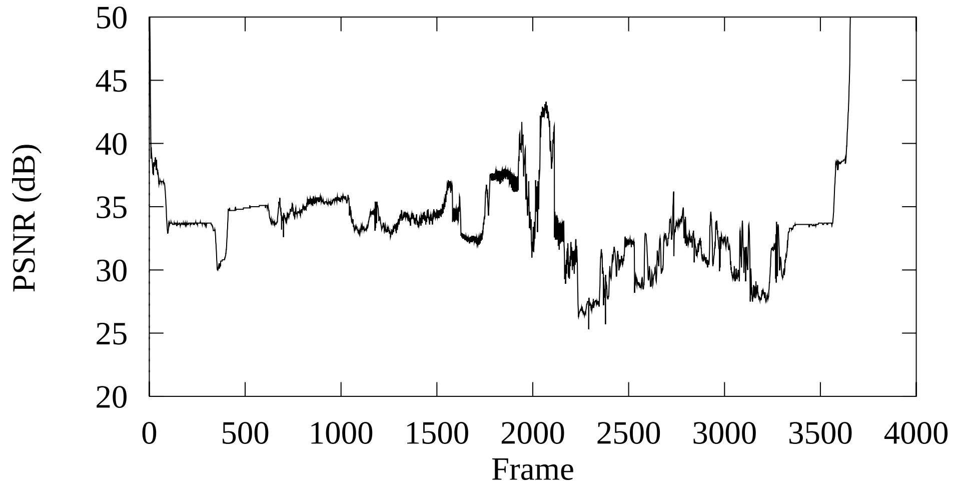


(c) Lexicographic VBR

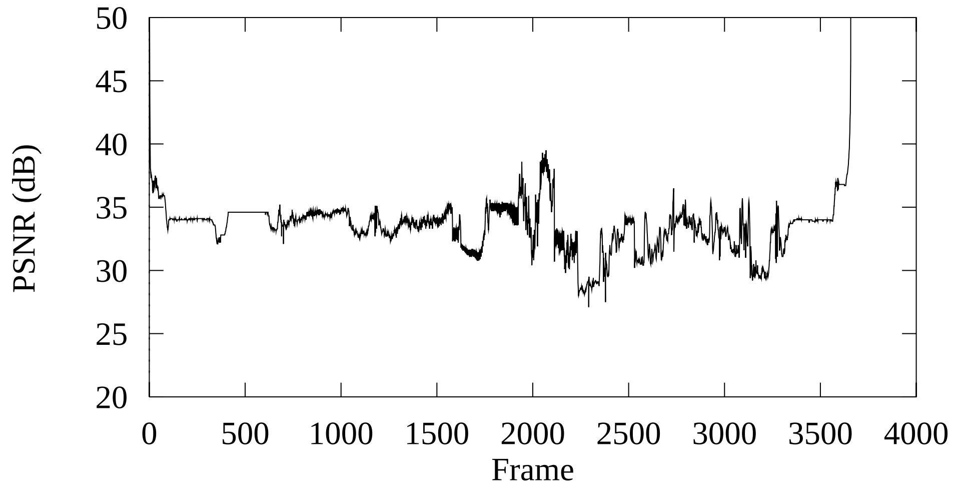
Figure 5: Nominal quantization scale for independent coding simulations.



(a) TM5 CBR



(b) Lexicographic CBR

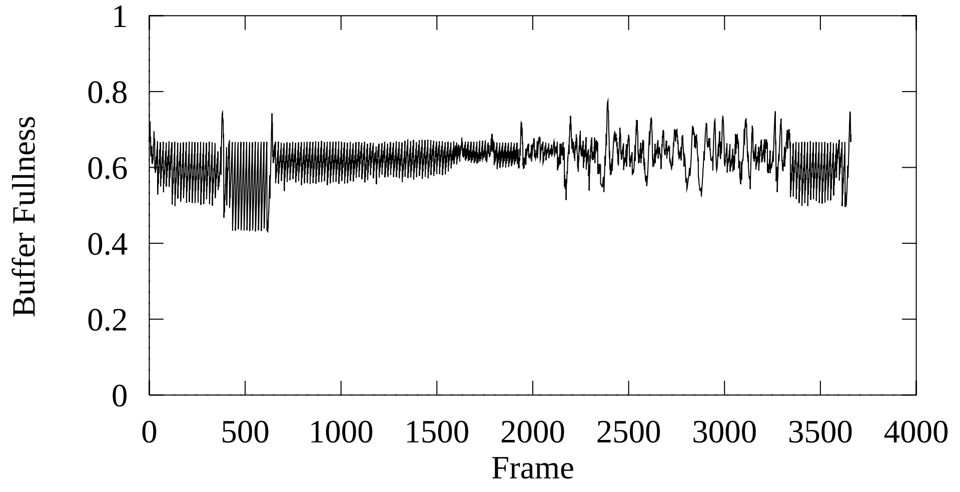


(c) Lexicographic VBR

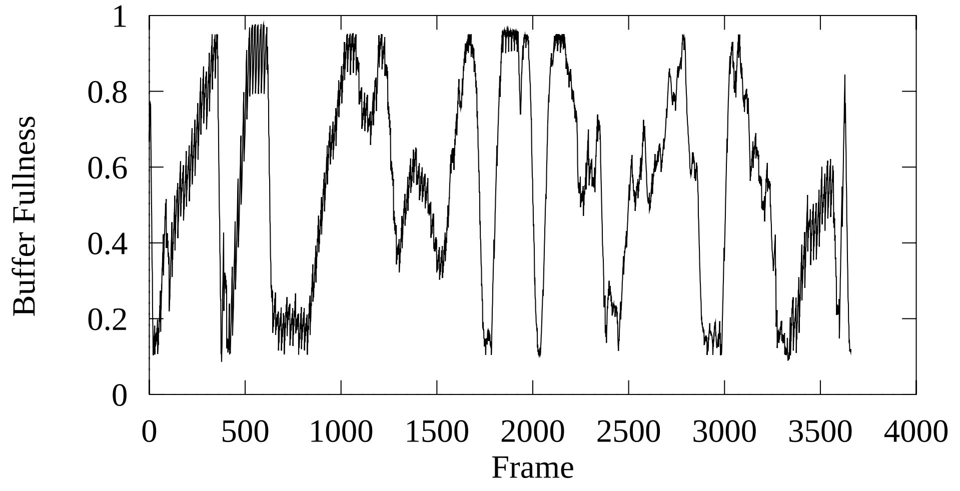
Figure 6: PSNR for independent coding simulations.

Table 2: Results of dependent coding simulations.

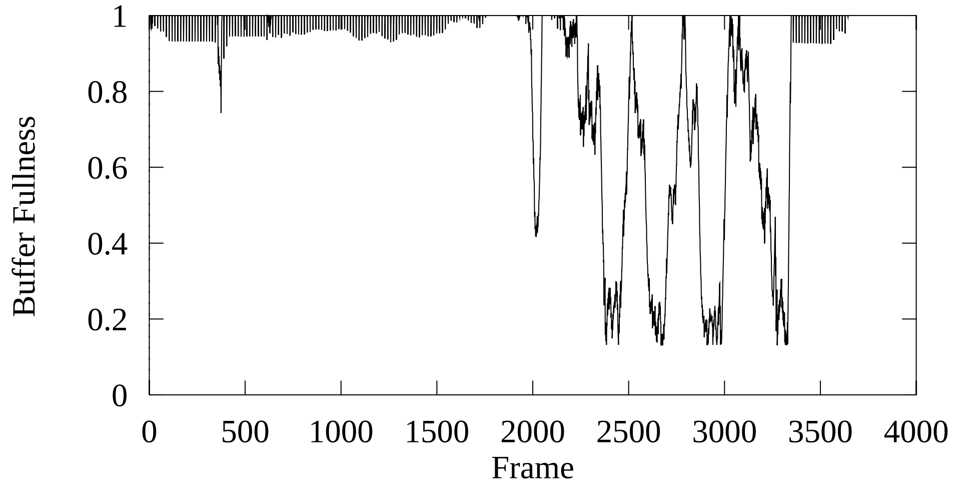
Method	Average PSNR (dB)	Std. Dev. of PSNR	Average Nom. Q	Std. Dev. of Nom. Q	Maximum Nom. Q	Minimum Nom. Q
TM5 CBR	33.34	4.95	14.10	10.47	52.57	1.93
Lexicographic CBR	33.45	4.75	13.04	7.96	29.01	2.25
Lexicographic VBR	33.09	2.54	11.74	1.97	16.84	9.02



(a) TM5 CBR

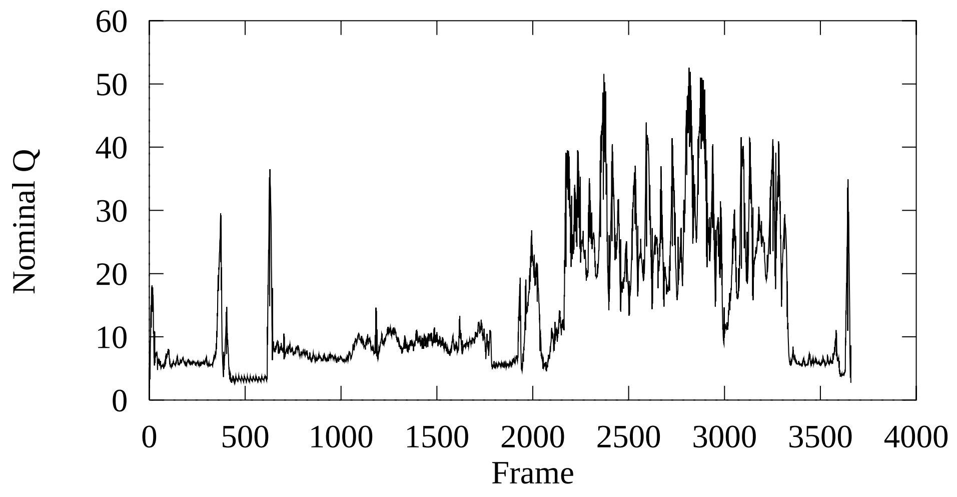


(b) Lexicographic CBR

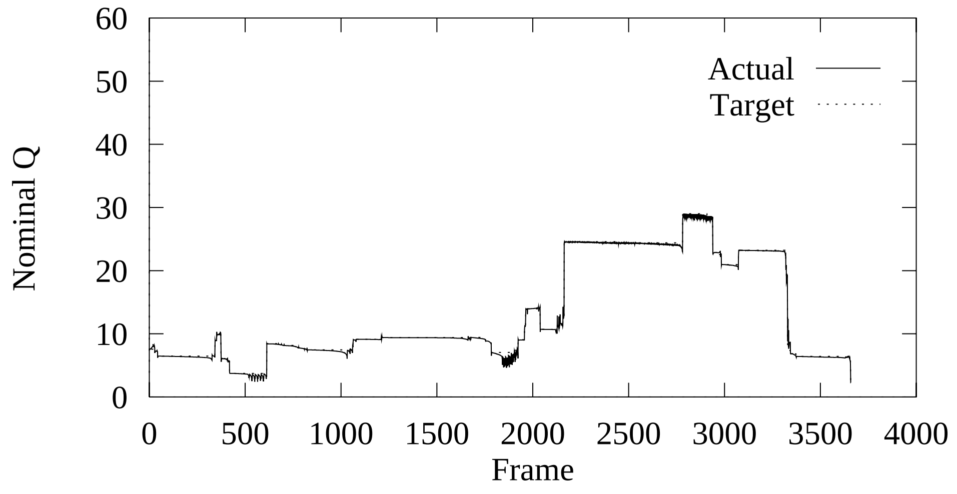


(c) Lexicographic VBR

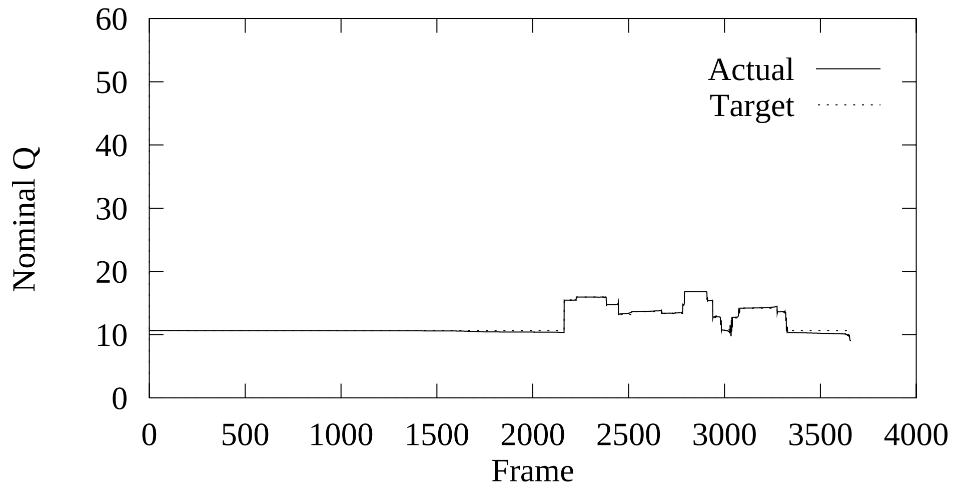
Figure 7: Evolution of normalized buffer fullness for dependent coding simulations.



(a) TM5 CBR

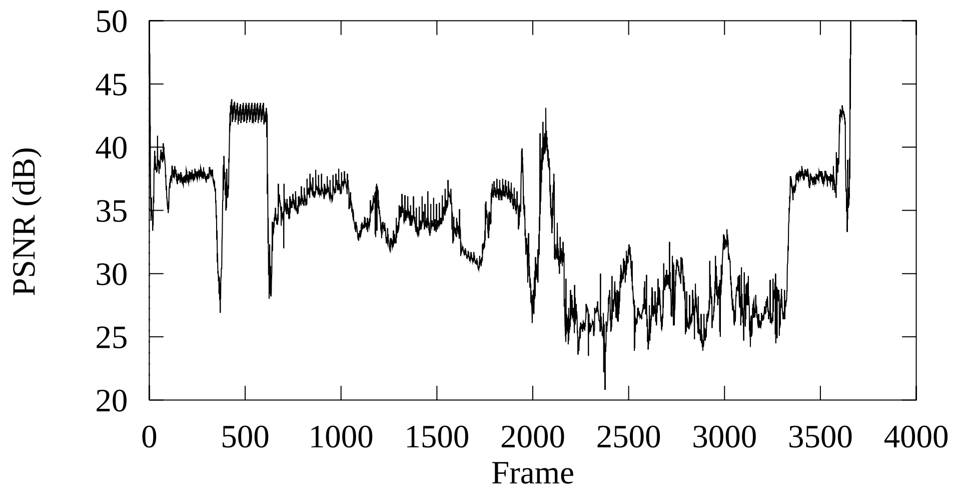


(b) Lexicographic CBR

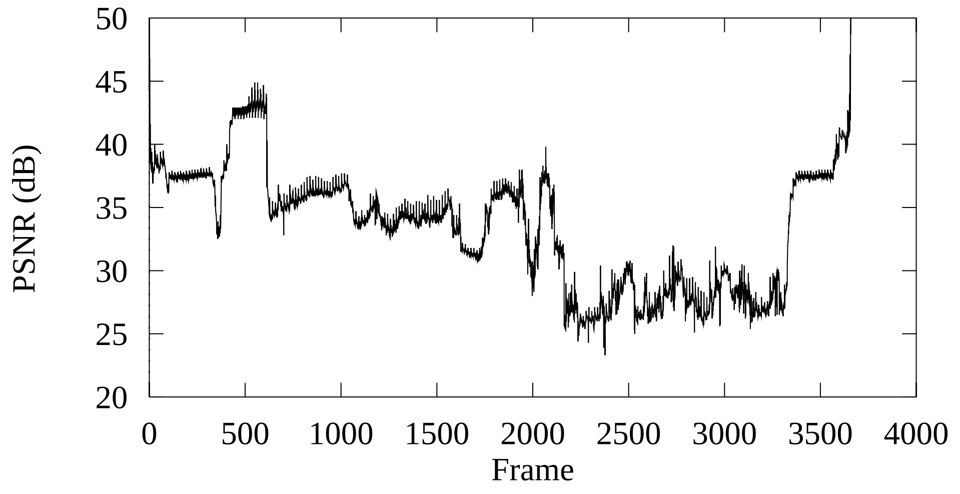


(c) Lexicographic VBR

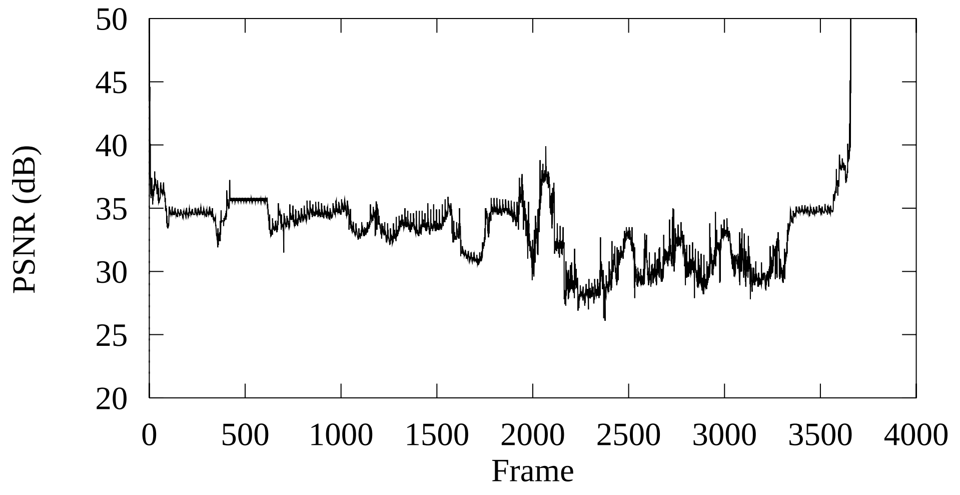
Figure 8: Nominal quantization scale for dependent coding simulations.



(a) TM5 CBR



(b) Lexicographic CBR



(c) Lexicographic VBR

Figure 9: PSNR for dependent coding simulations.