A Novel Lexicographic Framework for MPEG Rate Control

Dzung T. Hoang*
Digital Video Systems, Inc.
2710 Walsh Ave., Ste. 200
Santa Clara, CA 95051
dth@dvsystems.com

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

May 28, 2019

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.

Keywords

bit allocation, rate control, MPEG, video coding, lexicographic, lexicographic minimax, video compression

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 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, with an additional constraint on the total number of bits coded. 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.

Previous approaches in optimal rate control generally seeks to minimize a distortion measure, typically mean-squared error (MSE), averaged over coding blocks [1, 2]. While this approach leverages the wealth of

*Work was begun when the author was at IBM T. J. Watson Research Center. Support was provided in part by Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant F49620–94–1–0217

†Work was performed when author was at IBM T. J. Watson Research Center.

‡Support 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.

tools from optimization theory and operations research, it does not guarantee the constancy of 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 of 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 6, 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 6, we describe an implementation of these algorithms within a software MPEG-2 encoder and present simulation results.

2 Previous Work

In [3], the budget-constrained bit-allocation problem is examined 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 [4, 5]. 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, predictive coding setting is examined in [6]. 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.[1] The MINIMAX solution is obtained by equating the distortion among pictures.

In [7], bit-rate constraints for buffered video coders are derived for a general variable-bit-rate channel, such as that provided by an ATM network. The constraints take into account both the encoder and decoder buffers. The bit-rate constraints are used in an algorithm that jointly selects the channel and encoder rates.

The problem of optimal bit allocation in a buffered video coder is first considered in [8]. 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 Al-

[1] A MINIMAX solution is produced by our lexicographic framework since lexicographic optimality is a refinement of MINIMAX.

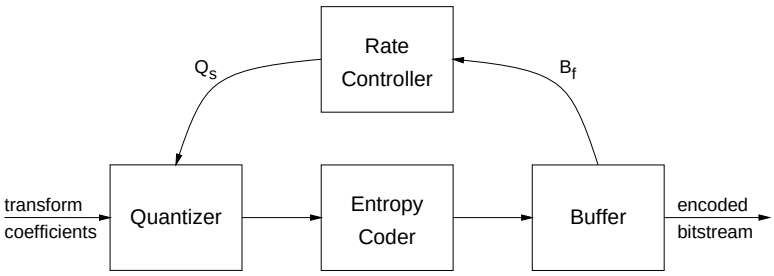


Figure 1: Block diagram of rate control in a typical video coding system.

gorithm. 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 [8] is extended in [9] 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.

In [10], the work in [8] is further extended 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 [11]. 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*.[2] 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 [12], 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 is applied to such problems as resource location and allocation (e.g., [13]). In this literature, lexicographic optimality is often referred to as *lexicographic minimax*, since it is viewed as a refinement of minimax theory.

3 Lexicographic Framework

3.1 Perceptual Quantization

As shown in Figure 1, the output bit rate of a video coder can be regulated by adjusting the 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

[2] This is similar to the nominal quantization scale defined in Section 3.1.

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) [14, 15]. These studies lead to various quantization techniques, called *perceptual quantization* or *adaptive quantization*, that take into account properties of the Human Visual System (HVS) in determining the quantization scale [16, 17, 18, 19, 20, 21, 22].

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 $Q_s = P(I, Q)$, 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 $P(I, Q) = \alpha_I Q$.[3] 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 perceptual quantization function.

The problem of determining $P(I, Q)$ has been studied elsewhere [16, 25] and is not considered 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 different Q for each block, this method results in constant perceptual quality within each picture given that perceptual quantization is employed at the block level. 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 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 $Q_s = \lfloor \alpha_I \cdot Q \rfloor$. 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 $0 \leq l_i < u_i$. (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 for coding each block. We assume that these choices are made independent of quantization and prior to performing rate control.

[3] The MPEG-2 Test Model 5 [23] also uses a multiplicative formulation while an additive formulation is proposed in [24].

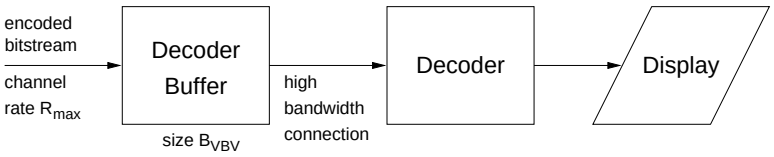


Figure 2: Block diagram of the MPEG Video Buffering Verifier.

3.3 Buffer 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), as shown in Figure 2. Data can be transferred to the VBV either at a constant or variable bit rate.[4] 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 $B_{\rm VBV}$ is specified in the bitstream by the encoder. The encoder also specifies the maximum transfer rate $R_{\rm max}$ 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 in each picture that follow from analysis of the VBV.

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 $R_{\rm max}$.

Definition 3.1 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 $B_{\rm VBV}$ be the size of the decoder buffer. Let $B_{\rm f}(s,n)$ denote the buffer fullness (the number of bits in the VBV buffer), resulting from allocation s, just *before* the nth picture is removed from the buffer. Let $B^*_{\rm f}(s,n)$ denote the buffer fullness, resulting from allocation s, just *after* the nth picture is removed. Then

$$B^*_{\rm f}(s,n) = B_{\rm f}(s,n) - s_n. \tag{1}$$

Let $R_{\rm max}$ be the rate at which bits enter the decoding buffer. Let T_n be the amount of time required to display picture n. Then $B_{\rm a}(n) = R_{\rm max}T_n$ is the maximum number of bits that can 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

$$\begin{array}{rcl} B_{\rm f}(s,1) & = & B_1, \\ B_{\rm f}(s,n+1) & = & B_{\rm f}(s,n) + B_{\rm a}(n) - s_n, \end{array} \tag{2}$$

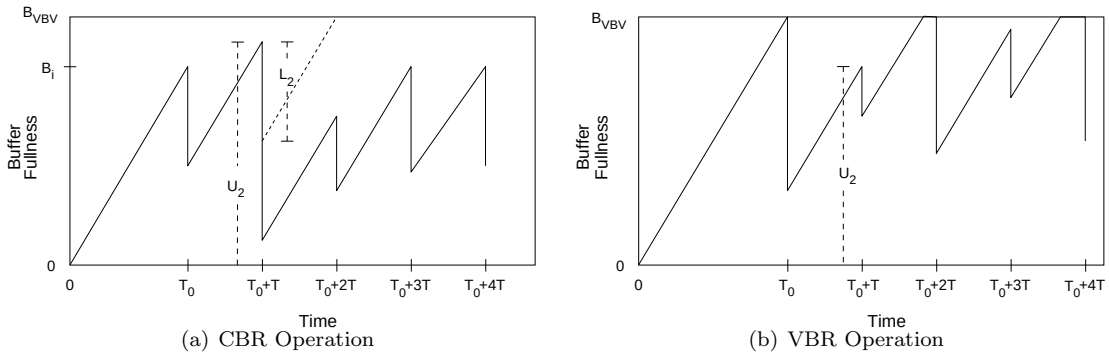
where B_1 is the initial buffer fullness. Unwinding the recurrence, we can also express (2) as

$$B_{\rm f}(s,n+1) = B_1 + \sum_{j=1}^{n} B_{\rm a}(j) - \sum_{j=1}^{n} s_j. \tag{3}$$

To prevent the decoder buffer from overflowing we must have

$$B_{\rm f}(s,n+1) \le B_{\rm VBV}. \tag{4}$$

[4]The MPEG-1 standard only defines VBV operation with a constant bit rate while the MPEG-2 standard also allows for variable bit rate.



(a) CBR Operation (b) VBR Operation

Figure 3: Sample plots of buffer fullness for CBR and VBR operation.

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

$$B_{\rm f}(s,n) \geq s_n, \tag{5}$$

or equivalently,

$$B^*_{\rm f}(s,n) \geq 0. \tag{6}$$

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 (2) and (4) we have

$$s_n \geq B_{\rm f}(s,n) + B_{\rm a}(n) - B_{\rm VBV}.$$

Since we cannot produce a negative number of bits, the lower bound on s_n is

$$s_n \geq \max\{B_{\rm f}(s,n) + B_{\rm a}(n) - B_{\rm VBV}, 0\}. \tag{7}$$

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

$$\max\{B_{\rm f}(s,n) + B_{\rm a}(n) - B_{\rm VBV}, 0\} \leq s_n \leq B_{\rm f}(s,n). \tag{8}$$

An exemplary plot of the evolution of the buffer fullness over time for CBR operation is shown in Figure 3(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 $R_{\max}$ 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 $R_{\max}$. The maximum number of bits entering the buffer in the time it takes to display picture n is $B_{\rm a}(n) = R_{\max}T_n$.

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

$$\begin{aligned} B_{\rm f}(s,1) &= B_{\rm VBV}, \\ B_{\rm f}(s,n+1) &= \min\{B_{\rm VBV}, B_{\rm f}(s,n) + B_{\rm a}(n) - s_n\}. \end{aligned} \tag{9}$$

Unlike the CBR case, the decoder buffer is prevented from overflowing by the minimization in (9). When $B_{\mathrm{f}}(s,n) + B_{\mathrm{a}}(n) - s_n > B_{\mathrm{VBV}}$, 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 (5) must hold.

In Figure 3(b), the evolution of the buffer fullness is shown for VBR operation. The time to display each picture is 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.

3.3.3 Encoder vs. Decoder Buffer

In the above discussion, we have focused solely on the decoder buffer whereas Figure 1 shows the Rate Controller monitoring the fullness of the encoder buffer. By assuming a fixed channel delay the encoder buffer fullness can be shown to mirror the decoder buffer fullness, except for an initial startup period. That is, an empty decoder buffer would correspond to a full encoder buffer, and vice versa. The reader is referred to [7] for a more complete discussion of buffer constraints in video coder systems.

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

$$P = \langle N, F, B_{\mathrm{tgt}}, B_{\mathrm{VBV}}, B_1, B_{\mathrm{a}} \rangle ,$$

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_{tgt} is the target number of bits to code all N pictures; B_{VBV} 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.

For convenience, in the sequel we shall use the shorter term "bit-allocation problem" to refer to the buffer-constrained bit-allocation problem.

Definition 3.3 Given a bit-allocation problem $P = \langle N, F, B_{\mathrm{tgt}}, B_{\mathrm{VBV}}, B_1, B_{\mathrm{a}} \rangle$, an allocation s is a *legal allocation* if the following conditions hold:

1. $\sum_{j=1}^{N} s_j = B_{\mathrm{tgt}}$
2. Equation (5) holds: $B_{\mathrm{f}}(s,n) \geq s_n$.
3. For CBR only, (7) holds: $s_n \geq \max\{B_{\mathrm{f}}(s,n) + B_{\mathrm{a}}(n) - B_{\mathrm{VBV}}, 0\}$.

In order for a CBR bit-allocation problem to have a legal allocation, we must have

$$B_{\mathrm{VBV}} \geq \max_j B_{\mathrm{a}}(j). \tag{10}$$

Also, the buffer fullness at the end of the sequence must be within bounds. For an allocation s, from (1) and (3) we have

$$B_{\mathrm{f}}^*(s,N) = B_1 + \sum_{j=1}^{N-1} B_{\mathrm{a}}(j) - B_{\mathrm{tgt}}. \tag{11}$$

The bound on $B_{\mathrm{f}}^*(s,N)$ is thus

$$0 \leq B_{\mathrm{f}}^*(s,N) \leq B_{\mathrm{VBV}}. \tag{12}$$

From (11) and (12), we have the following CBR bounds on $B_{\rm tgt}$:

$$B_1 + \sum_{j=1}^{N-1} B_{\rm a}(j) - B_{\rm VBV} \le B_{\rm tgt} \le B_1 + \sum_{j=1}^{N-1} B_{\rm a}(j). \tag{13}$$

A VBR bit-allocation problem does not have a lower bound for the target bit rate $B_{\rm tgt}$ since the VBV does not impose a lower bound on the number of bits produced by each picture. The upper bound for $B_{\rm tgt}$ depends upon whether $\max_{1\le j\le N}\{B_{\rm a}(j)\} > B_{\rm VBV}$. In general, the VBR upper bound on $B_{\rm tgt}$ is

$$B_{\rm tgt} \le B_1 + \sum_{j=1}^{N-1} \min\{B_{\rm a}(j), B_{\rm VBV}\}. \tag{14}$$

However, in the sequel, we assume that $\max_j\{B_{\rm a}(j)\} \le B_{\rm VBV}$. We also 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* in the literature (e.g., [26]).

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 $q_{j_1} \ge q_{j_2} \ge \cdots \ge q_{j_N}$. 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 ,..., $k-1$ and $\mathrm{rank}(s,k) > \mathrm{rank}(s',k)$ for some $1 \le k \le N$. We define $s \prec s'$ if and only if $s' \succ s$. We also define $s \asymp s'$ if and only if $\mathrm{rank}(s,j) = \mathrm{rank}(s',j)$ for all j. Similarly we define $s \succeq s'$ 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 3.4 A legal allocation s^* is *lexicographically optimal* if $s^* \preceq s$ for all other legal allocation s.

Lemma 3.1 (Constant-Q) *Given a bit-allocation problem $P = \langle N, F, B_{\rm tgt}, B_{\rm VBV}, B_1, B_{\rm a}\rangle$, 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

$$\sum_{j=1}^{N} s_j = \sum_{j=1}^{N} f_j(q) = B_{\rm tgt}.$$

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 $\mathrm{rank}(s',j) \le \mathrm{rank}(s,j)$ for all j. Therefore $s'_l > f_l(q)$ for some l and $s'_j \ge f_j(q)$ for all j since f_j is a decreasing function. Thus

$$\sum_{j=1}^{N} s'_j > \sum_{j=1}^{N} f_j(q) = B_{\rm tgt}.$$

So s' is not a legal allocation, 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 $\text{rank}(s, j) = \text{rank}(s', j)$ for all j. Therefore $\text{rank}(s', j) = q$ for all j. Thus $s' = s$. □

Lemma 3.1 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 and constant-Q) when it is feasible.

4 CBR Analysis and Algorithms

In this section, we analyze the buffer-constrained bit-allocation problem under constant-bit-rate VBV constraints, as described in Section 3.3.1. The analysis leads to an efficient dynamic programming algorithm for computing a lexicographically optimal solution.

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 seek to prove necessary conditions for lexicographic optimality. To do so, we need the following lemma.

Lemma 4.1 *Given two allocations s and s' of size N that satisfy $s_k = s'_k$ if and only if $k \notin \{u, v\}$, if $\max\{g_u(s'_u), g_v(s'_v)\} < \max\{g_u(s_u), g_v(s_v)\}$ then $s' \prec s$.*

Proof: Suppose $\max\{g_u(s'_u), g_v(s'_v)\} < \max\{g_u(s_u), g_v(s_v)\}$. Let j be the greatest index such that $\text{rank}(s, j) = \max\{g_u(s_u), g_v(s_v)\}$. Then $\text{rank}(s, j) > \text{rank}(s, j+1)$. Consider $\text{rank}(s', j)$. Either $\text{rank}(s', j) = \text{rank}(s, j+1)$ or $\text{rank}(s', j) = \max\{g_u(s'_u), g_v(s'_v)\}$. In either case, $\text{rank}(s, j) > \text{rank}(s', j)$. Therefore $s' \prec s$. □

The following lemma establishes necessary conditions for an optimal allocation.

Lemma 4.2 *Given a CBR bit-allocation problem $P = \langle N, F, B_{\text{tgt}}, B_{\text{VBV}}, B_1, B_{\text{a}} \rangle$, if s^* is an optimal allocation, the following are true:*

1. *If $g_j(s^*_j) > g_{j+1}(s^*_{j+1})$ for some $1 \le j < N$ then $B_{\text{f}}(s^*, j) = s^*_j$.*
2. *If $g_j(s^*_j) < g_{j+1}(s^*_{j+1})$ for some $1 \le j < N$ then $B_{\text{f}}(s^*, j+1) = B_{\text{VBV}}$.*

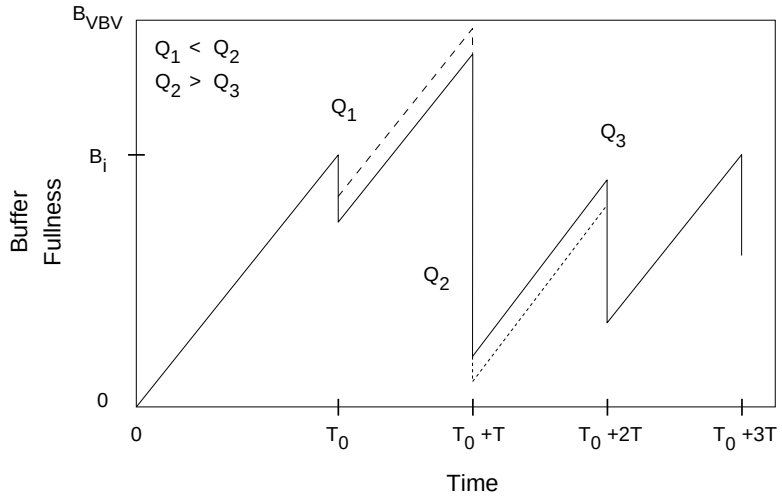


Figure 4: Sketch for proof of Lemma 4.2.

Lemma 4.2 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.

A sketch of the proof of Lemma 4.2 is shown in Figure 4. The proof is by contradiction. In the figure, the VBV buffer is shown for a hypothetical situation in which $Q_1 < Q_2$ and $Q_2 > Q_3$ and the switching conditions are not met.

In the first case, for $Q_1 < Q_2$, 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 and decreasing Q_2 while still maintaining $Q_1 < Q_2$ and not causing the buffer to overflow. This results in a better allocation than before. 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.

In the second case, for $Q_2 > Q_3$, 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 and increasing Q_3 while still maintaining $Q_2 > Q_3$ and not causing the buffer to underflow. This results in a better allocation than before. 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.

We note that Lemma 3.1 follows directly from Lemma 4.2.

Proof of Lemma 4.2:
Case 1. We prove Case 1 by contradiction. Suppose $B_{\rm f}(s^*, j) \neq s^*_j$. Let $\Delta = B_{\rm f}(s^*, j) - s^*_j$. Then by (5), $\Delta > 0$. Consider an allocation s' that differs from s^* only for pictures j and $j+1$; that is,

$$s'_k = s^*_k \quad \text{for } k \in \{1, \ldots, N\} \setminus \{j, j+1\} \qquad \text{and} \qquad s'_k \neq s^*_k \quad \text{for } k \in \{j, j+1\}\,. \tag{15}$$

In order to show a contradiction, we want to find an assignment to s'_j and s'_{j+1} that makes s' a legal allocation

and "better" than s^*. By "better" we mean

$$g_j(s'_j), g_{j+1}(s'_{j+1}) < g_j(s^*_j). \tag{16}$$

Equivalently, we want

$$s'_j > s^*_j \tag{17}$$

and

$$s'_{j+1} > f_{j+1}(g_j(s^*_j)). \tag{18}$$

To meet the target bit rate, we must have

$$s'_j + s'_{j+1} = s^*_j + s^*_{j+1}. \tag{19}$$

Let $\delta = s'_j - s^*_j$. Then $s^*_{j+1} - s'_{j+1} = \delta$. By (17), we want $\delta > 0$. We want to show that s' is a legal allocation for some value of $\delta > 0$. To avoid VBV violations, (8) must hold for all pictures under the allocation s'. From (15) and (19), we have

$$B_{\mathrm{f}}(s', k) = B_{\mathrm{f}}(s^*, k) \quad \text{for } k \neq j+1. \tag{20}$$

Since s^* is a legal allocation, there are no VBV violations for pictures 1, 2, ..., $j-1$ under s'. Furthermore, if our choice for s'_j does not cause a VBV violation for picture j, then we are assured that there would be no VBV violations in pictures $j+1$, $j+2$, ..., N. So we must choose s'_j subject to (8) and (17). Therefore

$$s^*_j < s'_j \leq s^*_j + \Delta. \tag{21}$$

If $0 < \delta \leq \Delta$, then s' is a legal allocation. We also want (18) to hold. For this we need

$$\delta < s^*_{j+1} - f_{j+1}(g_j(s^*_j)). \tag{22}$$

Since $g_j(s^*_j) > g_{j+1}(s^*_{j+1})$, we have $f_{j+1}(g_j(s^*_j)) < s^*_{j+1}$. Therefore $s^*_{j+1} - f_{j+1}(g_j(s^*_j)) > 0$. So for

$$0 < \delta \leq \min\{\Delta, s^*_{j+1} - f_{j+1}(g_j(s^*_j))\} \tag{23}$$

s' is a legal allocation that meets condition (16). By Lemma 4.1, $s^* \succ s'$ and s^* is not an optimal allocation, a contradiction.

Case 2. We prove Case 2 by contradiction. Suppose $B_{\mathrm{f}}(s^*, j+1) \neq B_{\mathrm{VBV}}$. Let $\Delta = B_{\mathrm{VBV}} - B_{\mathrm{f}}(s^*, j+1)$. Then by (4), $\Delta > 0$. Consider an allocation s' that differs from s^* only for pictures j and $j+1$; that is,

$$s'_k = s^*_k \text{ for } k \in \{1, \ldots, N\} \setminus \{j, j+1\}. \tag{24}$$

We want to find an assignment to s'_j and s'_{j+1} that makes s' a legal allocation and "better" than s^*, in order to show a contradiction. By "better" we mean

$$g_j(s'_j), g_{j+1}(s'_{j+1}) < g_{j+1}(s^*_{j+1}). \tag{25}$$

Equivalently, we want

$$s'_{j+1} > s^*_{j+1} \tag{26}$$

and

$$s'_j > f_j(g_{j+1}(s^*_{j+1})). \tag{27}$$

To meet the target bit rate, we must have

$$s'_j + s'_{j+1} = s^*_j + s^*_{j+1}. \tag{28}$$

Let $\delta = s'_{j+1} - s^*_{j+1}$. Then $s^*_j - s'_j = \delta$. By (26), we want $\delta > 0$. We want to show that s' is a legal allocation for some value of $\delta > 0$. To avoid VBV violations, (8) must hold for all pictures under the allocation s'. From (24) and (28), we have

$$B_{\mathrm{f}}(s', k) = B_{\mathrm{f}}(s^*, k) \quad \text{for } k \neq j+1. \tag{29}$$

Since s^* is a legal allocation, there are no VBV violations for pictures 1 to $j-1$ under s'. Furthermore, if our choice for s'_j does not cause a VBV violation, then we are assured that there would be no VBV violations in pictures $j+1$ to N. So we must choose s'_j subject to (8) and (26).

$$\begin{aligned} B_{\rm f}(s', j+1) &= B_{\rm f}(s', j) + B_{\rm a}(j) - s'_j \\ B_{\rm f}(s', j+1) &= B_{\rm f}(s^*, j) + B_{\rm a}(j) - s'_j \\ B_{\rm f}(s^*, j+1) &= B_{\rm f}(s^*, j) + B_{\rm a}(j) - s^*_j \\ B_{\rm f}(s', j+1) &= B_{\rm f}(s^*, j+1) + s^*_j - s'_j \\ B_{\rm f}(s', j+1) &= B_{\rm VBV} - \Delta + \delta \le B_{\rm VBV} \end{aligned}$$

From the above, we require $\delta \le \Delta$. If $0 < \delta \le \Delta$, then s' is a legal allocation. We also want (27) to hold. For this we need

$$\delta < s^*_j - f_j(g_{j+1}(s^*_{j+1})). \tag{30}$$

Since $g_{j+1}(s^*_{j+1}) > g_j(s^*_j)$, we have $f_j(g_{j+1}(s^*_{j+1})) < s^*_j$. Therefore $s^*_j - f_j(g_{j+1}(s^*_{j+1})) > 0$. So for

$$0 < \delta \le \min\{\Delta, s^*_j - f_j(g_{j+1}(s^*_{j+1}))\} \tag{31}$$

s' is a legal allocation that meets condition (27). By Lemma 4.1, $s^* \succ s'$ and s^* is not an optimal allocation, a contradiction.

□

The theorem that follows is the main result of this section and shows that the switching conditions are also sufficient for optimality. But first we prove a useful lemma that will be helpful in the proof of the theorem.

Lemma 4.3 *Given bit-allocations s and s' with $s_j \le s'_j$ for $u \le j \le v$ and $B_{\rm f}(s,u) \ge B_{\rm f}(s',u)$, we have $B_{\rm f}(s, v+1) = B_{\rm f}(s', v+1)$ if and only if $B_{\rm f}(s,u) = B_{\rm f}(s',u)$ and $s_j = s'_j$ for $u \le j \le v$.*

Proof: We use (3) to express $B_{\rm f}(s, v+1)$ in terms of $B_{\rm f}(s,u)$.

$$\begin{aligned} B_{\rm f}(s, v+1) &= B_1 + \sum_{j=1}^{v} (B_{\rm a}(j) - s_j) \\ B_{\rm f}(s, u) &= B_1 + \sum_{j=1}^{u-1} (B_{\rm a}(j) - s_j) \\ B_{\rm f}(s, v+1) &= B_{\rm f}(s,u) + \sum_{j=u}^{v} (B_{\rm a}(j) - s_j)\,. \end{aligned}$$

Similarly,

$$B_{\rm f}(s', v+1) = B_{\rm f}(s', u) + \sum_{j=u}^{v} \left(B_{\rm a}(j) - s'_j\right).$$

First we prove the "if" part. Suppose $B_{\rm f}(s,u) = B_{\rm f}(s',u)$ and $s_j = s'_j$ for $u \le j \le v$. Then

$$\begin{aligned} B_{\rm f}(s, v+1) &= B_{\rm f}(s,u) + \sum_{j=u}^{v} (B_{\rm a}(j) - s_j) \\ &= B_{\rm f}(s',u) + \sum_{j=u}^{v} \left(B_{\rm a}(j) - s'_j\right) \\ &= B_{\rm f}(s', v+1). \end{aligned}$$

Now we prove the "only if" part. Suppose $B_{\rm f}(s, v+1) = B_{\rm f}(s', v+1)$. Then

$$\begin{aligned} B_{\rm f}(s, v+1) &= B_{\rm f}(s', v+1) \\ B_{\rm f}(s, u) + \sum_{j=u}^{v} (B_{\rm a}(j) - s_j) &= B_{\rm f}(s', u) + \sum_{j=u}^{v} \left(B_{\rm a}(j) - s'_j\right) \\ B_{\rm f}(s, u) - B_{\rm f}(s', u) &= \sum_{j=u}^{v} (s_j - s'_j). \end{aligned} \tag{32}$$

But $B_{\rm f}(s, u) \geq B_{\rm f}(s', u)$ and $s_j \leq s'_j$ for $u \leq j \leq v$. Therefore $B_{\rm f}(s, u) - B_{\rm f}(s', u) \geq 0$ and $\sum_{j=u}^{v} (s_j - s'_j) \leq 0$. Combined with (32), this implies that $B_{\rm f}(s, u) = B_{\rm f}(s', u)$ and $\sum_{j=u}^{v} s_j = \sum_{j=u}^{v} s'_j$. Since $s_j \leq s'_j$ for $u \leq j \leq v$, this implies that $s_j = s'_j$ for $u \leq j \leq v$. □

Theorem 4.1 *Given a CBR bit-allocation problem $P = \langle N, F, B_{\rm tgt}, B_{\rm VBV}, B_1, B_{\rm a} \rangle$, a legal allocation s is optimal if and only if the following conditions hold. Also, the optimal allocation is unique.*

1. *If $g_j(s_j) > g_{j+1}(s_{j+1})$ for some $1 \leq j < N$, then $B_{\rm f}(s, j) = s_j$.*
2. *If $g_j(s_j) < g_{j+1}(s_{j+1})$ for some $1 \leq j < N$, then $B_{\rm f}(s, j+1) = B_{\rm VBV}$.*

Proof: Lemma 4.2 established these conditions as necessary for optimality. Now we need to show that these conditions are also sufficient for optimality and imply uniqueness. Let s be a legal allocation that meets both conditions of the theorem.

Let s^* be an optimal allocation for P. Let $Q_{\max} = \max_{1 \leq j \leq N} \{g_j(s_j)\}$. Consider the segments of consecutive pictures that are assigned the maximum Q by allocation s. Let u be the index of the start of such a segment. There are two cases: $u = 1$ and $u > 1$. If $u = 1$, then $B_{\rm f}(s, u) = B_{\rm f}(s^*, u) = B_1$. If $u > 1$, then since u is the index of the start of the segment, $g_{u-1}(s_{u-1}) < g_u(s_u)$ which implies that $B_{\rm f}(s, u) = B_{\rm VBV}$ by condition 2. Since s^* is a legal allocation, $B_{\rm f}(s^*, u) \leq B_{\rm VBV} = B_{\rm f}(s, u)$. In either case we have

$$B_{\rm f}(s, u) \geq B_{\rm f}(s^*, u). \tag{33}$$

Let v be the index of the end of the segment. Since s^* is optimal, $g_j(s^*_j) \leq Q_{\max}$ for all j, and thus

$$s^*_j \geq s_j \quad \text{for } u \leq j \leq v. \tag{34}$$

Therefore

$$B_{\rm f}(s^*, j) \leq B_{\rm f}(s, j) \quad \text{for } u \leq j \leq v. \tag{35}$$

There are two cases for v: $v = N$ and $v < N$. If $v = N$, then $B_{\rm f}(s, v+1) = B_{\rm f}(s^*, v+1) = B_1 + \sum_{j=1}^{N-1} B_{\rm a}(j) - B_{\rm tgt}$. If $v < N$, then since v is the index of the end of the segment, $g_v(s_v) > g_{v+1}(s_{v+1})$ which implies that

$$B_{\rm f}(s, v) = s_v \tag{36}$$

by condition 1. Since s^* is a legal allocation, $B_{\rm f}(s^*, v) \geq s^*_v$. Combine this with (36) and (34) we have

$$B_{\rm f}(s^*, v) \geq B_{\rm f}(s, v). \tag{37}$$

Combining (35) and (37), we have $B_{\rm f}(s^*, v) = B_{\rm f}(s, v)$ and $s^*_v = s_v$. As a result, $B_{\rm f}(s^*, v+1) = B_{\rm f}(s, v+1)$. In either case, $B_{\rm f}(s^*, v+1) = B_{\rm f}(s, v+1)$. By Lemma 4.3, $B_{\rm f}(s^*, u) = B_{\rm f}(s, u)$ and $s^*_j = s_j$ for $u \leq j \leq v$. As a consequence, $B_{\rm f}(s, j) = B_{\rm f}(s^*, j)$ for $u \leq j \leq v$.

We partition pictures 1 through N into segments, where, according to allocation s, the pictures in a segment use the same value of Q, either the first picture in the segment is picture 1 or the first picture in the segment uses a value of Q different from the previous picture, and either the last picture in the segment is picture N or the last picture in the segment uses a value of Q different from the next picture. Let M be

the number of such segments. We order the segments so that segment k uses a value of Q greater than or equal to the value of Q used in segment j for $j < k$, and denote the value of Q used by segment k as $Q_k^{(s)}$.

We will show that allocation s^* uses the same number of bits as allocation s for each picture in segment k, for $1 \le k \le M$. This will establish the conditions in the theorem as necessary and show that the optimal allocation is unique. We will prove this by induction on k. The base case of $k = 1$ has already been proven. *Inductive Hypothesis*: For all segments of pictures u to v with $u = 1$ or $g_{u-1}(s_{u-1}) \neq g_u(s_u)$, $v = N$ or $g_v(s_v) \neq g_{v+1}(s_{v+1})$, and $g_j(s_j) = Q_k^{(s)}$, we have $s_j^* = s_j$ and $B_\mathrm{f}(s, u) = B_\mathrm{f}(s^*, u)$ for $u \le j \le v$.

Let us assume that the hypothesis is true for $1 \le k < m$. We will show that is is also true for $k = m$. Consider a segment of consecutive pictures that are assigned quantization $Q_m^{(s)}$ by allocation s. Let u be the index of the start of the segment and v the index of the end of the segment.

By the inductive hypothesis, s and s^* use the same values of Q for all pictures for which s uses $Q > Q_m^{(s)}$. Because s^* is optimal, $g_j(s_j^*) \le g_j(s_j) = Q_m^{(s)}$ for $u \le j \le v$, and thus

$$s_j^* \ge s_j \quad \text{for } u \le j \le v. \tag{38}$$

We consider all cases for the segment boundaries. For the left segment boundary there are three cases: $u = 1$, $g_{u-1}(s_{u-1}) > g_u(s_u)$, and $g_{u-1}(s_{u-1}) < g_u(s_u)$. If $u = 1$, then $B_\mathrm{f}(s^*, u) = B_\mathrm{f}(s, u) = B_1$. If $g_{u-1}(s_{u-1}) > g_u(s_u)$, then from the inductive hypothesis, we have $B_\mathrm{f}(s^*, u-1) = B_\mathrm{f}(s, u-1)$ and $s_{u-1}^* = s_{u-1}$; therefore $B_\mathrm{f}(s^*, u) = B_\mathrm{f}(s, u)$. If $g_{u-1}(s_{u-1}) < g_u(s_u)$, then from condition 2, we have $B_\mathrm{f}(s, u) = B_\mathrm{VBV}$; since s^* is a legal allocation, $B_\mathrm{f}(s^*, u) \le B_\mathrm{VBV} = B_\mathrm{f}(s, u)$. For all three cases we have

$$B_\mathrm{f}(s^*, u) \le B_\mathrm{f}(s, u). \tag{39}$$

For the right segment boundary there are three cases: $v = N$, $g_v(s_v) < g_{v+1}(s_{v+1})$, and $g_v(s_v) > g_{v+1}(s_{v+1})$. If $v = N$, then $B_\mathrm{f}(s^*, v+1) = B_\mathrm{f}(s, v+1) = B_1 + \sum_{j=1}^{N-1} B_\mathrm{a}(j) - B_\mathrm{tgt}$. If $g_v(s_v) < g_{v+1}(s_{v+1})$, then from the inductive hypothesis, we have $B_\mathrm{f}(s^*, v+1) = B_\mathrm{f}(s, v+1)$.

If $g_v(s_v) > g_{v+1}(s_{v+1})$, then from condition 1 we have $B_\mathrm{f}(s, v) = s_v$. From (38) and (39) we have

$$B_\mathrm{f}(s^*, j) \le B_\mathrm{f}(s, j) \quad \text{for } u \le j \le v. \tag{40}$$

Since s^* is a legal allocation,

$$B_\mathrm{f}(s^*, v) \ge s_v^* \ge s_v = B_\mathrm{f}(s, v). \tag{41}$$

Combining (40) and (41), we have $B_\mathrm{f}(s^*, v) = B_\mathrm{f}(s, v)$ and $s_v^* = s_v$. Therefore $B_\mathrm{f}(s^*, v+1) = B_\mathrm{f}(s, v+1)$.

For all three cases we have

$$B_\mathrm{f}(s^*, v+1) = B_\mathrm{f}(s, v+1). \tag{42}$$

From (38), (39), (42), and Lemma 4.3, we have $s_j^* = s_j$ for $u \le j \le v$. It follows that $B_\mathrm{f}(s, j) = B_\mathrm{f}(s^*, j)$ for $u \le j \le v$. By induction, we have $s_j^* = s_j$ for all j, and so $s = s^*$. □

4.2 CBR Allocation Algorithm

Theorem 4.1 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 this section, we use the technique of dynamic programming (DP) to develop an algorithm to compute a lexicographically optimal CBR allocation in polynomial time and linear space.

4.2.1 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 4.1 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

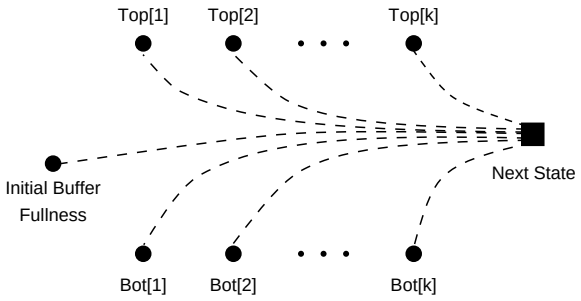


Figure 5: Illustration of search step in dynamic programming algorithm.

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 $\{\emptyset, \text{Top}^1, \ldots, \text{Top}^k, \text{Bot}^1, \ldots, \text{Bot}^k\}$, 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, $B_{\text{f}}(s, k+1) = B_{\text{VBV}}$. 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, $B_{\text{f}}(s, k+1) = s_{k+1}$.

The basic step in the DP algorithm is illustrated in Figure 5. 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_{tgt}.

4.2.2 Correctness of DP Algorithm

When computing Top^k and Bot^k for $1 \le k \le N-1$, we have insured that the conditions of Theorem 4.1 are met. Additionally in the final computation, the conditions are also met. Therefore we end up with a legal allocation that meets the conditions of Theorem 4.1 and is thus optimal.

4.2.3 Constant-Q Segments

We use 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:

$$F_{i,j}(q) = \sum_{i \le k \le j} f_k(q). \tag{43}$$

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.

4.2.4 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 set of pictures, say 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 $i \leq k < j$. More formally,

$$G_{i,j}(B_{\mathrm{T}}) \leq \min_{i \leq k < j} \Big\{ G_{i,k}\Big(B_1 + \sum_{i \leq m \leq k} B_{\mathrm{a}}(m) - B_{\mathrm{VBV}} \Big) \Big\}. \tag{44}$$

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 $i \leq k < j$. More formally,

$$G_{i,j}(B_{\mathrm{T}}) \geq \max_{i \leq k < j} \Big\{ G_{i,k}\Big(B_1 + \sum_{i \leq m < k} B_{\mathrm{a}}(m) \Big) \Big\}. \tag{45}$$

We can use these observations to perform all the VBV verifications in constant amortized time per verification.

4.2.5 Time and Space Complexity

The time complexity of the DP algorithm depends upon two main factors: the time it takes to evaluate f_i and $G_{i,j}$, and the time it takes to verify whether a sub-allocation is legal. If we assume that $G_{i,j}$ can be computed and the constant-Q allocation verified in time linear in the number of pictures involved, then we can show that to compute Top^k and Bot^k takes $O(k^2)$ time. This can be done for a large number of functional forms of the bit-production model f_i. In such cases, to compute an optimal allocation for a sequence of N pictures would take

$$\sum_{k=1}^{N} O(k^2) = O(N^3)$$

time. If we store pointers for tracing the optimal sequence of concatenations, the algorithm requires $O(N)$ space.

For some special cases of the bit-production model f_i, we can compute $G_{i,j}$ in constant time for each invocation with $O(N)$ preprocessing time and space. An example is $f_i(q) = \alpha_i/q + \beta_i$, where

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

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. Examples of other functional forms for f_i with a closed-form solution for $G_{i,j}$ can be found in [27]. Since a VBV verification can be done in constant amortized time, using these special forms for the bit-production model, we can compute an optimal CBR allocation for a sequence of N pictures in $O(N^2)$ time and $O(N)$ space.

5 VBR Analysis and Algorithms

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 [28]. In this application, statistical properties of the multiple video sequences allow more VBR bitstreams with a given peak rate $R_{\max}$ to be multiplexed onto the channel than CBR bitstreams coded at a constant rate of $R_{\max}$.

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 will be coded with an average bit rate that is 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.

5.1 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 5.1 *Given a VBR bit-allocation problem $P = \langle N, F, B_{\text{tgt}}, B_{\text{VBV}}, B_1, B_{\text{a}} \rangle$ and an optimal allocation s^*, if $B_{\text{f}}(s^*, j) + B_{\text{a}}(j) - s^*_j > B_{\text{VBV}}$ for $1 \le j \le N$, then $g_j(s^*_j) = \min_{1 \le k \le N}\{g_k(s^*_k)\}$.*

Proof: Let $Q_{\min} = \min_{1 \le k \le N}\{g_k(s^*_k)\}$. Let j be an index such that $B_{\text{f}}(s^*, j) + B_{\text{a}}(j) - s^*_j > B_{\text{VBV}}$. Let $\Delta = B_{\text{f}}(s^*, j) + B_{\text{a}}(j) - s^*_j - B_{\text{VBV}}$. Thus, $\Delta > 0$.

Suppose for the sake of contradiction that $g_j(s^*_j) > Q_{\min}$. Let u be an index such that $g_u(s^*_u) = Q_{\min}$. Consider an allocation s that differs from s^* only for pictures j and u. We want to assign values to s_j and s_u that make s a legal allocation with $g_j(s_j), g_u(s_u) < g_j(s^*_j)$, from which $s^* \succ s$, thereby arriving at a contradiction.

The idea is that we want to shift a (positive) number of bits, say δ, from picture u to picture j but still have a legal allocation. Let $s_j = s^*_j + \delta$ and $s_u = s^*_u - \delta$ with $\delta > 0$. Then $g_j(s_j) < g_j(s^*_j)$ and $\sum_{k=1}^{N} s_k = \sum_{k=1}^{N} s^*_k = B_{\text{tgt}}$. We now need to show that s does not result in a VBV buffer underflow, that is, $B_{\text{f}}(s, k) \ge s_k$ for $1 \le k \le N$. There are two cases to consider: $u < j$ and $u > j$.

Case 1: $u < j$. Since $s_k = s^*_k$ for $k < u$, we have $B_{\text{f}}(s, k) = B_{\text{f}}(s^*, k)$ for $1 \le k \le u$. Since $s_u < s^*_u$ and $s_k = s^*_k$ for $u < k < j$, we have $B_{\text{f}}(s, k) \ge B_{\text{f}}(s^*, k)$ for $u < k \le j$. Therefore pictures 1 to $j-1$ cannot cause any VBV buffer underflows. If we choose $0 < \delta < \Delta$, then $B_{\text{f}}(s, j+1) = B_{\text{VBV}}$ and picture j also cannot

cause a VBV buffer underflow. Since $s_k = s^*_k$ for $k > j$ and $B_{\rm f}(s, j+1) = B_{\rm f}(s^*, j+1)$, pictures $j+1$ to N also cannot cause any VBV buffer underflows.

Case 2: $u > j$**.** Since $s_k = s^*_k$ for $k < j$, we have $B_{\rm f}(s,k) = B_{\rm f}(s^*,k)$ for $1 \le k \le j$. If we choose $0 < \delta < \Delta$, then $B_{\rm f}(s, j+1) = B_{\rm VBV}$ and picture j also cannot cause a VBV buffer underflow. Since $s_k = s^*_k$ for $j < k < u$, and $B_{\rm f}(s, j+1) = B_{\rm f}(s^*, j+1)$ (by a suitable choice of δ), pictures $j+1$ to $u-1$ also cannot cause any VBV buffer underflows. Since $s_u < s^*_u$, we have $B_{\rm f}(s,k) \ge B_{\rm f}(s^*,k)$ for $k : k \ge u$. Therefore pictures u to N also cannot cause any VBV buffer underflows.

Therefore s is a legal allocation with $g_j(s_j) < g_j(s^*_j)$. We need to guarantee that $g_u(s_u) < g_j(s^*_j)$. Let $\gamma = g_j(s^*_j) - g_u(s^*_u)$. Since $g_j(s^*_j) > g_u(s^*_u)$, we have $\gamma > 0$. Let $\alpha = s^*_u - f_u(g_u(s^*_u) + \gamma/2)$. Since f_u is decreasing and $\gamma > 0$, we have $\alpha > 0$ and

$$\begin{aligned} s^*_u - \alpha &= f_u(g_u(s^*_u) + \gamma/2) \\ g_u(s^*_u - \alpha) &= g_u(s^*_u) + \gamma/2 \\ &< g_u(s^*_u) + \gamma \\ &= g_j(s^*_j). \end{aligned}$$

Consider the assignment $\delta = \min\{\alpha, \Delta/2\}$. There are two cases: $\alpha \le \Delta/2$ and $\alpha > \Delta/2$. If $\alpha \le \Delta/2$, we have $\delta = \alpha$ from which $g_u(s_u) = g_u(s^*_u - \delta) = g_u(s^*_u - \alpha) < g_j(s^*_j)$; since $0 < \delta < \Delta$, the allocation s is legal. If $\alpha > \Delta/2$, we have $\delta = \Delta/2$. Since g_u is decreasing and $\alpha > \Delta/2$, we have $g_u(s^*_u - \Delta/2) < g_u(s^*_u - \alpha)$ and thus $g_u(s_u) = g_u(s^*_u - \delta) = g_u(s^*_u - \Delta/2) < g_j(s^*_j)$. Since $0 < \delta < \Delta$, the allocation s is legal.

Since s is a legal allocation that differs from s^* only for pictures u and j with $g_u(s_u), g_j(s_j) < g_j(s^*_j)$, from Lemma 4.1 we have $s^* \succ s$, but s^* is not optimal, a contradiction. Therefore $g_j(s^*_j) = \min_{1 \le k \le N}\{g_k(s^*_k)\}$. □

Lemma 5.2 *Given a VBR bit-allocation problem* $P = \langle N, F, B_{\rm tgt}, B_{\rm VBV}, B_1, B_{\rm a}\rangle$ *and an optimal allocation* s^**, if* $B_{\rm f}(s^*, N) > s^*_N$ *then* $g_N(s^*_N) = \min_{1 \le k \le N}\{g_k(s^*_k)\}$.

Proof: Let $Q_{\min} = \min_{1 \le k \le N}\{g_k(s^*_k)\}$. Let j be an index such that $B_{\rm f}(s^*, j) + B_{\rm a}(j) - s^*_j > B_{\rm VBV}$. Let $\Delta = B_{\rm f}(s^*, N) - s^*_N$. Since $B_{\rm f}(s^*, N) > s^*_N$, we have $\Delta > 0$. Suppose that $g_N(s^*_N) > Q_{\min}$. Let u be an index such that $g_u(s^*_u) = Q_{\min}$. Now consider an allocation s that differs from s^* only for pictures u and N. We want to assign values to s_N and s_u that make s a legal allocation with $g_N(s_N), g_u(s_u) < g_N(s^*_N)$, from which $s^* \succ s$, thereby arriving at a contradiction. Let $s_N = s^*_N + \delta$ and $s_u = s^*_u - \delta$ with $\delta > 0$. Then $g_N(s_N) < g_N(s^*_N)$ and $\sum_{k=1}^{N} s_k = \sum_{k=1}^{N} s^*_k = B_{\rm tgt}$. We now need to show that s does not result in a VBV buffer underflow, that is, $B_{\rm f}(s,k) \ge s_k$ for $1 \le k \le N$.

Since $s_k = s^*_k$ for $k < u$, we have $B_{\rm f}(s,k) = B_{\rm f}(s^*,k)$ for $1 \le k \le u$. Since $s_u < s^*_u$ and $s_k = s^*_k$ for $u < k < N$, we have $B_{\rm f}(s,k) \ge B_{\rm f}(s^*,k)$ for $u < k \le N$. Therefore pictures 1 to $N-1$ cannot cause any VBV buffer underflows. For picture N, we have $B_{\rm f}(s,N) \ge B_{\rm f}(s^*,N) = \Delta + s^*_N = \Delta + s_N - \delta$. Therefore if we choose $\delta < \Delta$, then $B_{\rm f}(s,N) > s_N$ and picture N also cannot cause a VBV buffer underflow.

Therefore s is a legal allocation with $g_N(s_N) < g_N(s^*_N)$. We need to guarantee that $g_u(s_u) < g_N(s^*_N)$. Let $\gamma = g_N(s^*_N) - g_u(s^*_u)$. Since $g_N(s^*_N) > g_u(s^*_u)$, we have $\gamma > 0$. Let $\alpha = s^*_u - f_u(g_u(s^*_u) + \gamma/2)$. Since f_u is decreasing and $\gamma > 0$, we have $\alpha > 0$ and

$$\begin{aligned} s^*_u - \alpha &= f_u(g_u(s^*_u) + \gamma/2) \\ g_u(s^*_u - \alpha) &= g_u(s^*_u) + \gamma/2 \\ &< g_u(s^*_u) + \gamma \\ &= g_N(s^*_N). \end{aligned}$$

Consider the assignment $\delta = \min\{\alpha, \Delta/2\}$. There are two cases: $\alpha \le \Delta/2$ and $\alpha > \Delta/2$. If $\alpha \le \Delta/2$, we have $\delta = \alpha$ from which $g_u(s_u) = g_u(s^*_u - \delta) = g_u(s^*_u - \alpha) < g_N(s^*_N)$; since $0 < \delta < \Delta$, the allocation s is legal. If $\alpha > \Delta/2$, we have $\delta = \Delta/2$. Since g_u is decreasing and $\alpha > \Delta/2$, we have $g_u(s^*_u - \Delta/2) < g_u(s^*_u - \alpha)$ and thus $g_u(s_u) = g_u(s^*_u - \delta) = g_u(s^*_u - \Delta/2) < g_N(s^*_N)$. Since $0 < \delta < \Delta$, the allocation s is legal.

Since s is a legal allocation that differs from s^* only for pictures u and N with $g_u(s_u), g_N(s_N) < g_N(s^*_N)$, from Lemma 4.1, we have $s^* \succ s$, and s^* is not optimal, a contradiction. Therefore $g_N(s^*_N) = \min_{1 \le k \le N}\{g_k(s^*_k)\}$. □

The next lemma gives a set of *switching* conditions for changes in Q that are similar to the results of Lemma 4.2.

Lemma 5.3 *Given a VBR bit-allocation problem $P = \langle N, F, B_{\text{tgt}}, B_{\text{VBV}}, B_1, B_{\text{a}} \rangle$, if s^* is an optimal allocation, the following are true:*

1. *If $g_j(s_j^*) > g_{j+1}(s_{j+1}^*)$ for $1 \le j < N$, then $B_{\text{f}}(s^*, j) = s_j^*$.*
2. *If $g_j(s_j^*) < g_{j+1}(s_{j+1}^*)$ for $1 \le j < N$, then $B_{\text{f}}(s^*, j+1) = B_{\text{VBV}}$ and $B_{\text{f}}(s^*, j+1) + B_{\text{a}}(j+1) - s_{j+1}^* \le B_{\text{VBV}}$.*

Proof:
Case 1. The proof is identical to the proof of Case 1 of Lemma 4.2, except that condition (5) now holds instead of (8).
Case 2. Suppose that Case 2 is false. Then either $B_{\text{f}}(s^*, j+1) < B_{\text{VBV}}$ or $B_{\text{f}}(s^*, j+1) + B_{\text{a}}(j+1) - s_{j+1}^* > B_{\text{VBV}}$. Suppose that $B_{\text{f}}(s^*, j+1) + B_{\text{a}}(j+1) - s_{j+1}^* > B_{\text{VBV}}$. Then by Lemma 5.1, $g_{j+1}(s_{j+1}^*) \le g_j(s_j^*)$, a contradiction. Therefore $B_{\text{f}}(s^*, j+1) + B_{\text{a}}(j+1) - s_{j+1}^* \le B_{\text{VBV}}$.

Suppose that $B_{\text{f}}(s^*, j+1) < B_{\text{VBV}}$. Let $\Delta = B_{\text{VBV}} - B_{\text{f}}(s^*, j+1)$. Then $\Delta > 0$. Consider an allocation s that differs from s^* only for pictures j and $j+1$. We want to assign values to s_j and s_{j+1} that make s a legal allocation with $g_j(s_j), g_{j+1}(s_{j+1}) < g_{j+1}(s_{j+1}^*)$, from which $s^* \succ s$, thereby arriving at a contradiction.

Let $\gamma = g_{j+1}(s_{j+1}^*) - g_j(s_j^*)$ and $\alpha = s_j^* - f_j(g_j(s_j^*) + \gamma/2)$. Since $g_j(s_j^*) < g_{j+1}(s_{j+1}^*)$, we have $\gamma > 0$. Since f_j is decreasing and $\gamma > 0$, we have $\alpha > 0$ and

$$\begin{aligned} s_j^* - \alpha &= f_j(g_j(s_j^*) + \gamma/2) \\ g_j(s_j^* - \alpha) &= g_j(s_j^*) + \gamma/2 \\ &< g_j(s_j^*) + \gamma \\ &= g_{j+1}(s_{j+1}^*). \end{aligned}$$

Consider the assignments $s_j = s_j^* - \delta$ and $s_{j+1} = s_{j+1}^* + \delta$, where $\delta = \min\{\alpha, \Delta/2\}$. By this assignment, we have $g_{j+1}(s_{j+1}) < g_{j+1}(s_{j+1}^*)$. We now show that $g_j(s_j) < g_{j+1}(s_{j+1}^*)$. There are two cases: $\alpha \le \Delta/2$ and $\alpha > \Delta/2$. If $\alpha \le \Delta/2$, we have $\delta = \alpha$ from which $g_j(s_j) = g_j(s_j^* - \delta) = g_j(s_j^* - \alpha) < g_{j+1}(s_{j+1}^*)$. If $\alpha > \Delta/2$, we have $\delta = \Delta/2$. Since g_j is decreasing and $\alpha > \Delta/2$, we have $g_j(s_j^* - \Delta/2) < g_j(s_j^* - \alpha)$ and thus $g_j(s_j) = g_j(s_j^* - \delta) = g_j(s_j^* - \Delta/2) < g_{j+1}(s_{j+1}^*)$. In either case, we have $g_j(s_j) < g_{j+1}(s_{j+1}^*)$.

We now need to show that allocation s as defined above is a legal allocation. Since $s_k = s_k^*$ for $k < j$, we have $B_{\text{f}}(s, k) = B_{\text{f}}(s^*, k)$ for $1 \le k \le j$. Therefore there are no VBV buffer violations in pictures 1 to $j-1$. Since $s_j < s_j^*$, we have $B_{\text{f}}(s, j+1) > B_{\text{f}}(s^*, j+1)$. Therefore picture j cannot cause a VBV buffer underflow.

Now we need to show that pictures $j+1$ to N also cannot cause a VBV buffer underflow. Since $B_{\text{f}}(s^*, j+1) < B_{\text{VBV}}$, we have $B_{\text{f}}(s^*, j+1) = B_{\text{f}}(s^*, j) + B_{\text{a}}(j) - s_j^*$ and

$$\begin{aligned} B_{\text{f}}(s, j) + B_{\text{a}}(j) - s_j &= B_{\text{f}}(s^*, j) + B_{\text{a}}(j) - (s_j^* - \delta) \\ &= B_{\text{f}}(s^*, j) + B_{\text{a}}(j) - s_j^* + \delta \\ &= B_{\text{f}}(s^*, j+1) + \delta \\ &< B_{\text{f}}(s^*, j+1) + \Delta \\ &= B_{\text{VBV}}. \end{aligned}$$

Thus $B_{\text{f}}(s, j+1) = B_{\text{f}}(s, j) + B_{\text{a}}(j) - s_j$. We have already shown that $B_{\text{f}}(s^*, j+1) + B_{\text{a}}(j+1) - s_{j+1}^* \le B_{\text{VBV}}$. Therefore $B_{\text{f}}(s^*, j+2) = B_{\text{f}}(s^*, j+1) + B_{\text{a}}(j+1) - s_{j+1}^*$. Now,

$$\begin{aligned} B_{\text{f}}(s, j+1) + B_{\text{a}}(j+1) - s_{j+1} &= B_{\text{f}}(s, j) + B_{\text{a}}(j) - s_j + B_{\text{a}}(j+1) - s_{j+1} \\ &= B_{\text{f}}(s^*, j+1) + \delta + B_{\text{a}}(j+1) - (s_{j+1}^* + \delta) \\ &= B_{\text{f}}(s^*, j+1) + B_{\text{a}}(j+1) - s_{j+1}^* \\ &= B_{\text{f}}(s^*, j+2) \\ &\le B_{\text{VBV}}. \end{aligned}$$

Therefore $B_{\rm f}(s, j+2) = B_{\rm f}(s^*, j+2)$. Since $s_k = s_k^*$ for $k > j+1$, we have $B_{\rm f}(s,k) = B_{\rm f}(s^*,k)$ for $k > j+1$. Therefore pictures $j+1$ to N cannot cause a VBV buffer underflow and s is a legal allocation.

Since s is a legal allocation that differs from s^* only for pictures j and $j+1$ and we have $g_j(s_j), g_{j+1}(s_{j+1}) < g_{j+1}(s_{j+1}^*)$, from Lemma 4.1 we have $s^* \succ s$, but s^* is not optimal, a contradiction.

□

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

Theorem 5.1 *Given a VBR bit-allocation problem $P = \langle N, F, B_{\rm tgt}, B_{\rm VBV}, B_1, B_{\rm a}\rangle$, a legal allocation s is optimal if and only if the following conditions hold. Also, the optimal allocation is unique.*

1. *If $B_{\rm f}(s,j) + B_{\rm a}(j) - s_j > B_{\rm VBV}$ for $1 \le j \le N$, then $g_j(s_j) = \min_{1\le k\le N}\{g_k(s_k)\}$.*
2. *If $B_{\rm f}(s^*, N) > s_N^*$ then $g_N(s_N^*) = \min_{1\le k\le N}\{g_k(s_k^*)\}$.*
3. *If $g_j(s_j) > g_{j+1}(s_{j+1})$ for $1 \le j < N$, then $B_{\rm f}(s,j) = s_j$.*
4. *If $g_j(s_j) < g_{j+1}(s_{j+1})$ for $1 \le j < N$, then $B_{\rm f}(s, j+1) = B_{\rm VBV}$ and $B_{\rm f}(s,j+1) + B_{\rm a}(j+1) - s_{j+1} \le B_{\rm VBV}$.*

Proof: Lemmas 5.1, 5.2, and 5.3 establish these as necessary conditions. Now we need to show that these conditions are also sufficient for optimality and imply uniqueness.

The proof for sufficiency and uniqueness is similar to that of Theorem 4.1 except for segments with the minimum Q. Here we consider only segments that use the minimum Q.

Let s^* be an optimal allocation, $Q_{\min} = \min_{1\le j\le N}\{g_j(s_j)\}$, and $J_{\min} = \{j : g_j(s_j) = Q_{\min}\}$. By condition 2, if $g_N(s_N) > Q_{\min}$ then it must be that $B_{\rm f}(s,N) = s_N$, or equivalently, $B_{\rm f}(s, N+1) = B_{\rm a}(N)$. Therefore $B_{\rm f}(s,N)$ is known if picture N does not use the minimum Q, and we can use arguments of Theorem 4.1. Following the steps of Theorem 4.1, we can show that $s_j^* = s_j$ for $j : g_j(s_j) > Q_{\min}$.

Since s^* is optimal, we have $g_j(s_j^*) \le g_j(s_j)$ for $j \in J_{\min}$. Therefore

$$s_j^* \ge s_j \text{ for } j \in J_{\min}. \tag{46}$$

Since the total number of bits allocated is the same for s and s^*, we have the number of bits to be allocated to pictures in J must also be the same. That is,

$$\sum_{j\in J_{\min}} s_j^* = \sum_{j\in J_{\min}} s_j. \tag{47}$$

But (46) and (47) both hold if and only if $s_j^* = s_j$ for $j \in J_{\min}$. Therefore $s = s^*$. □

Although Theorem 5.1 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 5.4 *Given two VBR bit-allocation problems $P^{(1)} = \left\langle N, F, B_{\rm tgt}^{(1)}, B_{\rm VBV}, B_1, B_{\rm a}\right\rangle$ and $P^{(2)} = \left\langle N, F, B_{\rm tgt}^{(2)}, B_{\rm VBV}, B_1, B_{\rm a}\right\rangle$ that have optimal allocations $s^{(1)}$ and $s^{(2)}$, respectively, with $B_{\rm tgt}^{(1)} < B_{\rm tgt}^{(2)}$, then $s^{(1)} \succ s^{(2)}$.*

Proof: Let $J_{\rm over} = \left\{j : B_{\rm f}(s^{(1)}, j) + B_{\rm a}(j) - s_j^{(1)} > B_{\rm VBV}\right\}$. Then $J_{\rm over}$ contains exactly the pictures that result in virtual overflows, as defined in Section 3.3.2. If we start with allocation $s^{(1)}$, it is clear that we can use more bits for the pictures in $J_{\rm over}$ without changing the buffer fullness $B_{\rm f}(s^{(1)}, n)$. Let $B_{\rm over} = \sum_{j\in J_{\rm over}}\left(B_{\rm f}(s^{(1)}, j) + B_{\rm a}(j) - s_j^{(1)} - B_{\rm VBV}\right)$. Then $B_{\rm over}$ is the maximum number of bits we can add to the pictures in $J_{\rm over}$ without changing the buffer fullness. Let $\Delta = B_{\rm tgt}^{(2)} - B_{\rm tgt}^{(1)}$. There are two cases to consider: $\Delta \le B_{\rm over}$ and $\Delta > B_{over}$.

Case 1: $\Delta \le B_{\text{over}}$**.** Consider an allocation s for problem $P^{(2)}$ constructed as follows. Let $s_j = s_j^{(1)}$ for $j \notin J_{\text{over}}$. We then distribute Δ bits to the pictures in J_{over} without changing the buffer fullness. Then $s_j \ge s_j^{(1)}$ which implies that $g_j(s_j) \le g_j(s_j^{(1)})$. Since $\Delta > 0$, we also have $s_j > s_j^{(1)}$ for some $j \in J_{\text{over}}$. Since $B_{\text{f}}(s, j) = B_{\text{f}}(s^{(1)}, j)$ for all j, s does not cause any buffer underflows. Since we used $B_{\text{tgt}}^{(1)} + \Delta = B_{\text{tgt}}^{(2)}$ bits in s, s is a legal allocation for $P^{(2)}$.

Case 2: $\Delta > B_{\text{over}}$**.** Consider an allocation s for problem $P^{(2)}$ constructed as follows. Let $s_j = s_j^{(1)}$ for $j \notin J_{\text{over}} \cup \{N\}$. We then distribute B_{over} bits to pictures in J_{over}. We do this with the assignments: $s_j = s^{(1)} + (B_{\text{f}}(s^{(1)}, j) + B_{\text{a}}(j) - s_j^{(1)} - B_{\text{VBV}})$ for $j \in J_{\text{over}}$. Finally, we distribute the remaining $\Delta - B_{\text{over}}$ bits to picture N with $s_N = s_N^{(1)} + \Delta - B_{\text{over}}$.

We have shown how to create a legal allocation s for $P^{(2)}$ starting with $s^{(1)}$. When we add more bits to $s^{(1)}$ to form s, we strictly decrease Q for the pictures that we add bits to and never increase Q anywhere. Therefore $s^{(1)} \succ s$. Since $s^{(2)}$ is the optimal allocation for $P^{(2)}$, we have $s \succeq s^{(2)}$. Therefore $s^{(1)} \succ s^{(2)}$.

□

Lemma 5.5 *Given two VBR bit-allocation problems* $P^{(1)} = \left\langle N, F, B_{\text{tgt}}^{(1)}, B_{\text{VBV}}, B_1, B_{\text{a}} \right\rangle$ *and* $P^{(2)} = \left\langle N, F, B_{\text{tgt}}^{(2)}, B_{\text{VBV}}, B_1, B_{\text{a}} \right\rangle$ *that have optimal allocations* $s^{(1)}$ *and* $s^{(2)}$*, respectively, with* $B_{\text{tgt}}^{(1)} < B_{\text{tgt}}^{(2)}$*, then* $s_j^{(1)} = s_j^{(2)}$ *for* j *such that* $g_j(s_j^{(1)}) > \min_{1 \le k \le N}\{g_k(s_k^{(1)})\}$.

Proof: We provide an inductive proof similar to that used to prove Theorem 4.1. First we assume that $s^{(1)}$ is not a constant-Q allocation, for if it were, the lemma would hold vacuously.

Let $Q_k^{(1)}$ be the kth largest value of Q assigned by allocation $s^{(1)}$. Let $Q_{\min}^{(1)}$ be the minimum value of Q. *Inductive Hypothesis*: For all segments of pictures u to v with $u = 1$ or $g_{u-1}(s_{u-1}) \ne g_u(s_u)$, $v = N$ or $g_v(s_v) \ne g_{v+1}(s_{v+1})$, and $g_j(s_j) = Q_k^{(1)} > Q_{\min}^{(1)}$, we have $s_j^{(1)} = s_j^{(2)}$ and $B_{\text{f}}(s^{(1)}, j) = B_{\text{f}}(s^{(2)}, j)$ for $u \le j \le v$.

We first prove the base case of $k = 1$. Consider the segments of consecutive pictures that are assigned quantization $Q_1^{(1)}$ by allocation $s^{(1)}$. Let u be the index of the start of such a segment. We consider two cases: $u = 1$ and $u > 1$. If $u = 1$, then $B_{\text{f}}(s^{(1)}, u) = B_{\text{f}}(s^{(2)}, u) = B_1$. If $u > 1$, then since u is the index of the start of the segment, we have $g_{u-1}(s_{u-1}^{(1)}) < g_u(s_u^{(1)})$, which implies that $B_{\text{f}}(s^{(1)}, u) = B_{\text{VBV}}$ by Lemma 5.3; since $s^{(2)}$ is a legal allocation, we have $B_{\text{f}}(s^{(2)}, u) \le B_{\text{VBV}}$. In either case we have

$$B_{\text{f}}(s^{(1)}, u) \ge B_{\text{f}}(s^{(2)}, u) \tag{48}$$

Let v be the index of the end of the segment. We consider two cases: $v = N$ and $v < N$. If $v = N$, then by the contrapositive of Lemma 5.2, $B_{\text{f}}(s^{(1)}, v) = s_v^{(1)}$. (Here we use the condition that $Q_1^{(1)} > Q_{\min}^{(1)}$.) If $v < N$, then since v is the index of the end of the segment, we have $g_v(s_v^{(1)}) > g_{v+1}(s_{v+1}^{(1)})$, which implies that $B_{\text{f}}(s^{(1)}, v) = s_v^{(1)}$ by Lemma 5.3. In either case we have

$$B_{\text{f}}(s^{(1)}, v) = s_v^{(1)}. \tag{49}$$

From Lemma 5.4, we have $s^{(1)} \succ s^{(2)}$. Therefore $g_j(s_j^{(2)}) \le Q_1^{(1)}$ for all j and thus

$$s_j^{(1)} \le s_j^{(2)} \text{ for } u \le j \le v. \tag{50}$$

From (48) and (50) we have

$$B_{\text{f}}(s^{(1)}, j) \ge B_{\text{f}}(s^{(2)}, j) \quad \text{for } u \le j \le v. \tag{51}$$

Since $s^{(2)}$ is a legal allocation, we have

$$B_{\text{f}}(s^{(2)}, v) \ge s_v^{(2)} \ge s_v^{(1)} = B_{\text{f}}(s^{(1)}, v). \tag{52}$$

Combining (51) and (52), we have $B_{\rm f}(s^{(1)},v) = B_{\rm f}(s^{(2)},v)$ and $s_v^{(1)} = s_v^{(2)}$. Therefore $B_{\rm f}(s^{(1)},v+1) = B_{\rm f}(s^{(2)},v+1)$. Since $Q_1^{(1)} > Q_{\min}^{(1)}$, by the contrapositive of Lemma 5.2, we see that the buffer fullness for pictures u to v is updated as with CBR operation. Therefore we can use the results of Lemma 4.3, which implies that $B_{\rm f}(s_j^{(1)}) = B_{\rm f}(s^{(2)},j)$ and $s_j^{(1)} = s_j^{(2)}$ for $u \le j \le v$.

Let us assume that the inductive hypothesis is true for $1 \le k < m$. We need to show that it is also true for $k = m$ where $Q_m^{(1)} > Q_{\min}^{(1)}$. Consider a segment of consecutive pictures that are assigned quantization $Q_m^{(1)}$. Let u be the index of the start of the segment and v the index of the end of the segment. We consider all cases for the segment boundaries. For the left segment boundary we consider three cases: $u = 1$, $g_{u-1}(s_{u-1}^{(1)}) > g_u(s_u^{(1)})$, and $g_{u-1}(s_{u-1}^{(1)}) < g_u(s_u^{(1)})$. If $u = 1$, then we have $B_{\rm f}(s^{(1)},u) = B_{\rm f}(s^{(2)},u) = B_1$. If $g_{u-1}(s_{u-1}^{(1)}) > g_u(s_u^{(1)})$, then from the inductive hypothesis, we have $B_{\rm f}(s^{(1)},u-1) = B_{\rm f}(s^{(2)},u-1)$ and $s_{u-1}^{(1)} = s_{u-1}^{(2)}$; therefore $B_{\rm f}(s^{(1)},u) = B_{\rm f}(s^{(2)},u)$. If $g_{u-1}(s_{u-1}^{(1)}) < g_u(s_u^{(1)})$, then from Lemma 5.3, we have $B_{\rm f}(s^{(1)},u) = B_{\rm VBV}$; since $s^{(2)}$ is a legal allocation, we have $B_{\rm f}(s^{(2)},u) \le B_{\rm VBV} = B_{\rm f}(s^{(1)},u)$. For all three cases we have

$$B_{\rm f}(s^{(2)},u) \le B_{\rm f}(s^{(1)},u). \tag{53}$$

For the right segment boundary we consider three cases: $v = N$, $g_v(s_v^{(1)}) > g_{v+1}(s_{v+1}^{(1)})$, and $g_v(s_v^{(1)}) < g_{v+1}(s_{v+1}^{(1)})$. If $v = N$, then by the contrapositive of Lemma 5.2, $B_{\rm f}(s^{(1)},v) = s_v^{(1)}$. (We use the condition that $Q_m^{(1)} \ne Q_{\min}^{(1)}$.) If $g_v(s_v^{(1)}) > g_{v+1}(s_{v+1}^{(1)})$, then by Lemma 5.3, $B_{\rm f}(s^{(1)},v) = s_v^{(1)}$. If $g_v(s_v^{(1)}) < g_{v+1}(s_{v+1}^{(1)})$, then from the inductive hypothesis, we have $B_{\rm f}(s^{(1)},v+1) = B_{\rm f}(s^{(2)},v+1)$. For the first two cases, we have

$$B_{\rm f}(s^{(1)},v) = s_v^{(1)} \tag{54}$$

From Lemma 5.4, we have $s^{(1)} \succ s^{(2)}$. Therefore $g_j(s_j^{(2)}) \le Q_m^{(1)}$ for $u \le j \le v$ and thus

$$s_j^{(1)} \le s_j^{(2)} \text{ for } u \le j \le v. \tag{55}$$

From (53) and (55) we have

$$B_{\rm f}(s^{(1)},j) \ge B_{\rm f}(s^{(2)},j) \quad \text{for } u \le j \le v. \tag{56}$$

Since $s^{(2)}$ is a legal allocation, we have

$$B_{\rm f}(s^{(2)},v) \ge s_v^{(2)} \ge s_v^{(1)} = B_{\rm f}(s^{(1)},v). \tag{57}$$

Combining (56) and (57), we have $B_{\rm f}(s^{(1)},v) = B_{\rm f}(s^{(2)},v)$ and $s_v^{(1)} = s_v^{(2)}$. Therefore $B_{\rm f}(s^{(1)},v+1) = B_{\rm f}(s^{(2)},v+1)$.

So for all three cases for v, we have $B_{\rm f}(s^{(1)},v+1) = B_{\rm f}(s^{(2)},v+1)$.

Since $Q_n^{(1)} > Q_{\min}^{(1)}$, by the contrapositive of Lemma 5.2, we see that the buffer fullness for pictures u to v is updated the same as with CBR operation. Therefore we can use the results of Lemma 4.3, which implies that $B_{\rm f}(s_j^{(1)}) = B_{\rm f}(s^{(2)},j)$ and $s_j^{(1)} = s_j^{(2)}$ for $u \le j \le v$.

By induction, we have $s_j^{(1)} = s_j^{(2)}$ for all j such that $g_j(s^{(1)},j) > Q_{\min}^{(1)}$. □

Lemma 5.6 *Given two VBR bit-allocation problems* $P^{(1)} = \left\langle N, F, B_{\rm tgt}^{(1)}, B_{\rm VBV}, B_1, B_{\rm a} \right\rangle$ *and* $P^{(2)} = \left\langle N, F, B_{\rm tgt}^{(2)}, B_{\rm VBV}, B_1, B_{\rm a} \right\rangle$ *that have optimal allocations* $s^{(1)}$ *and* $s^{(2)}$*, respectively, with* $B_{\rm tgt}^{(1)} < B_{\rm tgt}^{(2)}$*, then* $\min_{1\le j\le N}\{g_j(s_j^{(1)})\} > \min_{1\le j\le N}\{g_j(s_j^{(2)})\}$.

Proof: Let $Q_{\min}^{(1)} = \min_{1\le j\le N}\{g_j(s_j^{(1)})\}$ and $Q_{\min}^{(2)} = \min_{1\le j\le N}\{g_j(s_j^{(2)})\}$. From Lemma 5.4 we have $s^{(1)} \succ s^{(2)}$. From Lemma 5.5 we have the only pictures that can be assigned different Q by $s^{(1)}$ and $s^{(2)}$ are those that are assigned quantization $Q_{\min}^{(1)}$ by $s^{(1)}$. But $s^{(1)} \succ s^{(2)}$ which implies that $s^{(2)}$ must assign to some picture a quantization lower than $Q_{\min}^{(1)}$. Therefore $Q_{\min}^{(1)} > Q_{\min}^{(2)}$. □

We summarize Lemmas 5.4, 5.5, and 5.6 with the following theorem.

Theorem 5.2 *Given two VBR bit-allocation problems* $P^{(1)} = \left\langle N, F, B^{(1)}_{\text{tgt}}, B_{\text{VBV}}, B_1, B_{\text{a}} \right\rangle$ *and* $P^{(2)} = \left\langle N, F, B^{(2)}_{\text{tgt}}, B_{\text{VBV}}, B_1, \right.$ *that have optimal allocations* $s^{(1)}$ *and* $s^{(2)}$*, respectively, with* $B^{(1)}_{\text{tgt}} < B^{(2)}_{\text{tgt}}$*, then*

1. $s^{(1)} \succ s^{(2)}$,
2. $s^{(1)}_j = s^{(2)}_j$ *for* j *such that* $g_j(s^{(1)}_j) > \min_{1 \le k \le N}\{g_k(s^{(1)}_k)\}$*, and*
3. $\min_{1 \le j \le N}\{g_j(s^{(1)}_j)\} > \min_{1 \le j \le N}\{g_j(s^{(2)}_j)\}$.

5.2 VBR Allocation Algorithm

Theorems 5.1 and 5.2 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 5.2 gives us an iterative way to find the minimum Q.

5.2.1 VBR Algorithm

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

1. Mark all pictures as *easy*. Let $B_{\text{easy}} \leftarrow B_{\text{tgt}}$.
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. the segment starts with the first picture. segment, reset the buffer fullness to empty and continue the simulation.
4. Allocate bits to each newly identified hard segment according to the CBR Algorithm, with a bit budget 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 $B_{\text{easy}} \leftarrow B_{\text{tgt}} - B_{\text{hard}}$.
6. If a new hard segment has been identified in Step 3, goto Step 2.

5.2.2 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 5.1.

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 3 and 4 of Theorem 5.1 hold. In order to show that the other conditions also hold, we first need to show that the CBR Algorithm does not assign a Q lower than the $Q_{\min}$ computed in Step 2 of the VBR Algorithm.

Lemma 5.7 *Let s be an 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*

$$\min_{i \le k \le j} \{g_k(s_k)\} \ge Q_{\min}.$$

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 $B_{\rm f}(s,u) = B_{\rm f}(s',u)$ 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 4.1, $B_{\rm f}(s,u-1) = s_{u-1}$. This implies that $B_{\rm f}(s,u) = B_{\rm a}(u-1)$. Since s' does not cause an underflow for picture $u-1$, $B_{\rm f}(s',u-1) \ge s'_{u-1}$, which implies that $B_{\rm f}(s',u) \ge B_{\rm a}(u-1)$. In either case, we have

$$B_{\rm f}(s',u) \ge B_{\rm f}(s,u). \tag{58}$$

We consider two cases for v: $v = j$ and $v < j$. If $v = j$, then $B_{\rm f}(s',v) < s'_v$ since an underflow occurs at picture j. Thus $B_{\rm f}(s',v+1) < B_{\rm a}(v)$. But since s is a legal allocation, $B_{\rm f}(s,v+1) \ge B_{\rm a}(v)$. 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 4.1, $B_{\rm f}(s,v+1) = B_{\rm VBV}$. Since s' does not cause virtual overflow, $B_{\rm f}(s',v+1) \le B_{\rm VBV}$. In either case,

$$B_{\rm f}(s',v+1) \le B_{\rm f}(s,v+1). \tag{59}$$

Expanding for $B_{\rm f}(s,u)$ and $B_{\rm f}(s,v+1)$ we have

$$B_{\rm f}(s,u) = B_1 + \sum_{k=1}^{u-1} B_{\rm a}(k) - \sum_{k=1}^{u-1} s_k, \tag{60}$$

$$B_{\rm f}(s,v+1) = B_1 + \sum_{k=1}^{v} B_{\rm a}(k) - \sum_{k=1}^{v} s_k. \tag{61}$$

Subtracting (61) from (60), canceling like terms, and rearranging, we have

$$\sum_{k=u}^{v} s_k = \sum_{k=u}^{v} B_{\rm a}(k) + B_{\rm f}(s,u) - B_{\rm f}(s,v+1). \tag{62}$$

The same manipulations with $B_{\rm f}(s',u)$ and $B_{\rm f}(s',v+1)$ yield

$$\sum_{k=u}^{v} s'_k = \sum_{k=u}^{v} B_{\rm a}(k) + B_{\rm f}(s',u) - B_{\rm f}(s',v+1). \tag{63}$$

Combining (58), (59), (62), and (63) we have

$$\sum_{k=u}^{v} s_k \le \sum_{k=u}^{v} s'_k. \tag{64}$$

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

$$F_{u,v}\Big(\min_{i \le k \le j}\{g_k(s_k)\}\Big) \le F_{u,v}(Q_{\min}). \tag{65}$$

Since $F_{u,v}$ is a monotonically decreasing function (see Section 4.2.3), we have

$$\min_{i \le k \le j} \{g_k(s_k)\} \ge Q_{\min}.$$

□

From Lemma 5.7, 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 and does not include the last picture if it does not cause a buffer underflow, conditions 1 and 2 of Theorem 5.1 also hold.

We are now ready to state the main result of this section.

Theorem 5.3 (Correctness of VBR Algorithm) *Each pass through the VBR Algorithm results in an allocation that is lexicographically optimal for the number of bits actually allocated.*

5.2.3 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.

Assuming a special form for the bit-production model that results in execution time of $O(N^2)$ for the CBR Algorithm, we now show that the VBR algorithm also executes in $O(N^2)$ time and uses $O(N)$ space.

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 $O(N^2)$.

We can defer actually invoking the CBR Algorithm in Step 4 of the VBR Algorithm until the end. This would avoid invoking the CBR Algorithm more than once for each hard picture. Let M be the number of hard segments found by the VBR Algorithm and L_i be the size of the ith segment. The time consumed by execution of the CBR Algorithm can be expressed as

$$T_{\mathrm{CBR}}(N) = \sum_{i=1}^{M} O(L_i^2) = O\Big(\sum_{i=1}^{M} L_i^2\Big). \tag{66}$$

Since $\sum_{i=1}^{M} L_i \le N$, we have

$$\sum_{i=1}^{M} L_i^2 \le \Big(\sum_{i=1}^{M} L_i\Big)^2 \le N^2.$$

Therefore the time complexity of the VBR Algorithm is $O(N^2)$, the same as that for the CBR Algorithm. For cases where there are relatively few hard segments, computing an optimal VBR allocation will likely be faster in practice than computing a CBR allocation. Furthermore, Theorem 5.3 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.

5.3 Discussion

It is satisfying to see that optimal VBR bit allocation is no more complex (and often can be simpler) than optimal CBR bit allocation. This result goes against the prevailing perception that VBR rate control is harder than CBR rate control.

However, we should note that the above complexity analysis is performed in the context of off-line global optimization. The vast majority of CBR video coders in operation today work in real-time mode without the luxury of lookahead processing. Since VBR coders can potentially give better quality for the same bit budget, they are targeted for quality-sensitive applications (such as encoding a Hollywood movie) where expensive off-line processing is a viable option. However, the above analysis does allow for "one-pass" VBR encoding. By substituting a real-time CBR algorithm for the optimal one invoked by the VBR Algorithm, we can construct a one-pass real-time VBR encoder. Though necessarily sub-optimal, the resulting coder would have complexity comparable to existing CBR coders.

6 Implementation

In this section, we describe an implementation of rate control using the lexicographically optimal bit allocation algorithms presented in Sections 4.2 and 5.2 within a publicly available software MPEG-2 encoder [29]. 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 with the approximate models.

6.1 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 activity 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 activity factor. For modeling purposes, we define the nominal quantization for a picture as the average of the product of the reference quantization scale and the buffer-feedback factor over all coded macroblocks.

6.2 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 4.2.5 and 5.2.3 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 $G_{i,j}$. In this section we examine two classes of models for which $G_{i,j}$ can be efficiently computed.

6.2.1 Hyperbolic Model

In [16], the following simple "hyperbolic" model forms the basis of an adaptive bit allocation algorithm:

$$f_i(q_i) = \frac{\alpha_i}{q_i} + \beta_i, \tag{67}$$

where α_i is associated with the complexity of coding picture i and β_i with the overhead for coding the picture. TM5 adopts a similar model where only the complexity term is used. With these adaptive 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.

The hyperbolic model is one of the simplest models to exhibit the monotonicity and concavity characteristic of rate-distortion functions.[5] Several instances of the hyperbolic model are plotted in Figure 6.

With the hyperbolic model, there is a simple closed-form expression for $G_{i,j}$:

$$G_{i,j}(b) = \frac{\sum_{i \le k \le j} \alpha_k}{b - \sum_{i \le k \le j} \beta_k}. \tag{68}$$

As previously discussed in Section 4.2.5, 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. This results in a time complexity of $O(N^2)$ for both optimal CBR and VBR allocation.

[5] Our framework only assumes monotonicity and not concavity.

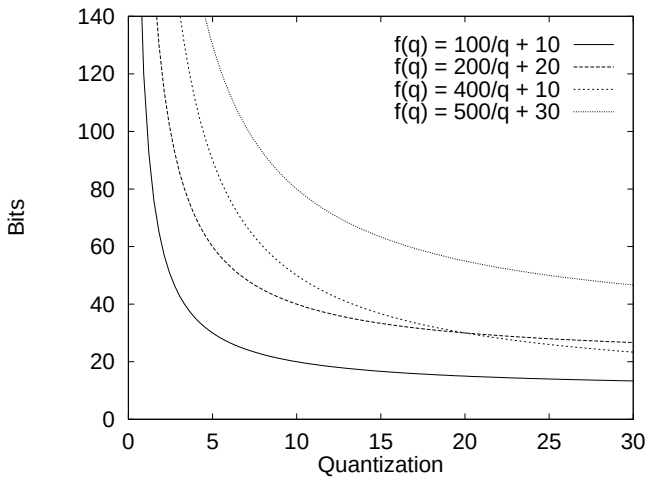


Figure 6: Several instances of a simple "hyperbolic" bit-production model.

In related work, Ding and Liu [30] propose the following more general class of bit-production models and describe its use in rate control:

$$f_i(q) = \frac{\alpha_i}{q^{\gamma_i}} + \beta_i. \tag{69}$$

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 (69) 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.

In preliminary experiments, we find that the hyperbolic model works well near the operating point where α_i and β_i have been determined, but is not reliable at a distant operating point. This observation leads us to formulate the following encoding strategy.

1. Encode the sequence using the standard TM5 coder, keeping statistics (for each picture) of the average nominal quantization scale, the total number of bits used, and the number of bits used to code the quantized DCT coefficients.

2. Compute α_i and β_i from the statistics gathered in the previous encoding pass. Allocate bits to pictures with the lexicographic bit allocation algorithm and encode the sequence using this allocation, gathering statistics as before.

3. Repeat Step 2.

The idea is that with each encoding, the accuracy of the bit models will improve as the operating Q is determined and refined for each picture.

6.2.2 Linear-Spline Model

As noted above, the hyperbolic model works well with small changes in the quantization scale Q. 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. Previously, we have compensated 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 to construct more accurate bit models that are then used to encode the video sequence in a single pass.

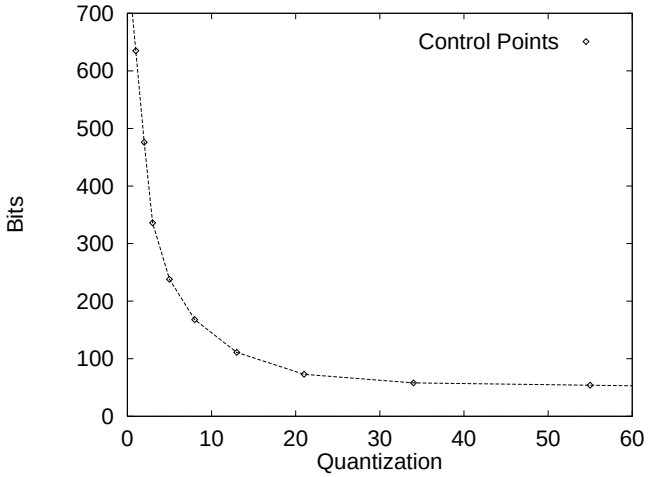


Figure 7: Example of a linear-spline interpolation model.

Lin, Ortega, and Kuo [31, 32] propose using cubic-spline interpolation models of rate and distortion in conjunction with a gradient-based rate control algorithm [33, 34]. The spline models are computed by first encoding each picture several times using a select set of M quantization scales, $\{x_1, \ldots, x_M\}$ with $x_1 < x_2 < \cdots < x_M$, and measuring the actual rate. Each quantization/rate pair is called a *control point*. For picture i, the function between two consecutive control points $(x_k, y_{i,k})$ and $(x_{k+1}, y_{i,k+1})$ has the form

$$f_i^k(x_k) = a_{ik}x^3 + b_{ik}x^2 + c_{ik}x + d_{ik}. \tag{70}$$

The parameters a_{ik}, b_{ik}, c_{ik}, and d_{ik} are computed from four control points $(x_{k-1}, y_{i,k-1})$, $(x_k, y_{i,k})$, $(x_{k+1}, y_{i,k+1})$, and $(x_{k+2}, y_{i,k+2})$, such that $f_i^k(x_k) = y_{i,k}$ and $f_i^{k+1}(x_{k+1}) = y_{i,k+1}$ and the first-order derivatives of f_i^k and f_i^{k+1} are continuous at the control points. 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 not monotonic, in general. 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 $(x_k, y_{i,k})$ and $(x_{k+1}, y_{i,k+1})$ has the form

$$f_i^k(x_k) = \alpha_{i,k}x + \beta_{i,k}. \tag{71}$$

Similar to [33, 34], we choose the control quantization scales to be $\{1, 2, 3, 5, 8, 13, 21, 34, 55\}$ to exploit the exponential-decay property of rate-distortion functions. In case the control points themselves do not exhibit monotonicity, we enforce monotonicity by skipping those control points where the monotonicity property is violated. For quantization scales less than x_1 or greater than x_M, we extrapolate using the parameters $(\alpha_{i,k}, \beta_{i,k})$ or $(\alpha_{i,M-1}, \beta_{i,M-1})$, respectively. An example of a linear-spline model is shown in Figure 7.

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

$$G_{i,j}(b) = \frac{b - \sum_{i \le m \le j} \beta_{m,k}}{\sum_{i \le m \le j} \alpha_{m,k}}. \tag{72}$$

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. A simple binary search procedure can be used to find the operating point.

Since there are a fixed number of control points, we can compute $G_{i,j}$ in constant time with linear-time preprocessing. As with the hyperbolic model, we can compute optimal CBR and VBR allocations in quadratic time.

The cubic-spline model of [31, 32] is used in a dependent-coding framework, where the effects of coding previous pictures are taken into account in the modeling. Our framework assumes independent coding and does not take these effects into account. However, from the switching theorems, we note that an optimal allocation has segments of constant Q. This provides a basis for estimating the linear-spline model parameters. By encoding the video sequence multiple times with a constant Q determined from the control points, we can construct a linear-spline interpolation model for each picture. We expect these models to be reasonably accurate within a segment of constant Q. At the boundary between segments, however, we can expect some discrepancy in the models for dependent pictures (P and B types).

6.3 Picture-Level Rate Control

Even with accurate bit-production models, the actual number of bits produced will inevitably depart from the model. There are essentially two ways to cope with bit-modeling errors.

6.3.1 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.

6.3.2 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 and potentially cause the buffer to overflow or underflow.

After coding a picture, we can reallocate bits to the remaining pictures optimally (for the given models). 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 4.2, 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.

With a reverse dynamic programming table, 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 for the hyperbolic and linear-spline interpolation models. The total additional time to

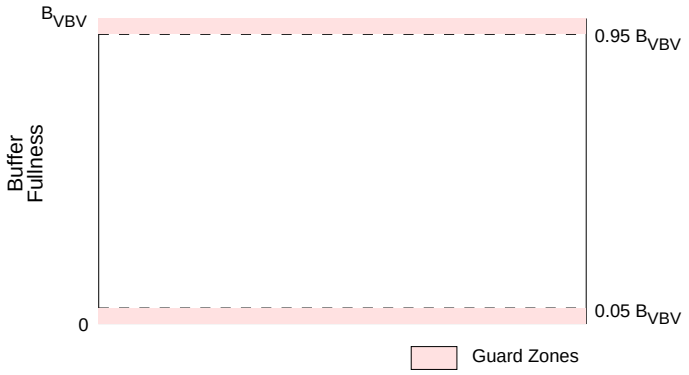


Figure 8: Guard zones to safeguard against underflow and overflow of VBV buffer. A bit allocation is computed so that the buffer fullness remains between the guard zones.

recover from bit-production errors is then $O(N^2)$, the same as the time complexity for computing the initial allocation.

As stated, the above procedure applies to a CBR allocation and to hard pictures in a VBR allocation (which are allocated using the CBR routine). 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.

6.3.3 Hybrid Rate Control

In early experiments, we observed that closed-loop rate control resulted in rapid fluctuations in the nominal quantization scale[6] 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. As noted earlier, since we assume independent coding, we can expect more errors in the bit-production models at 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.

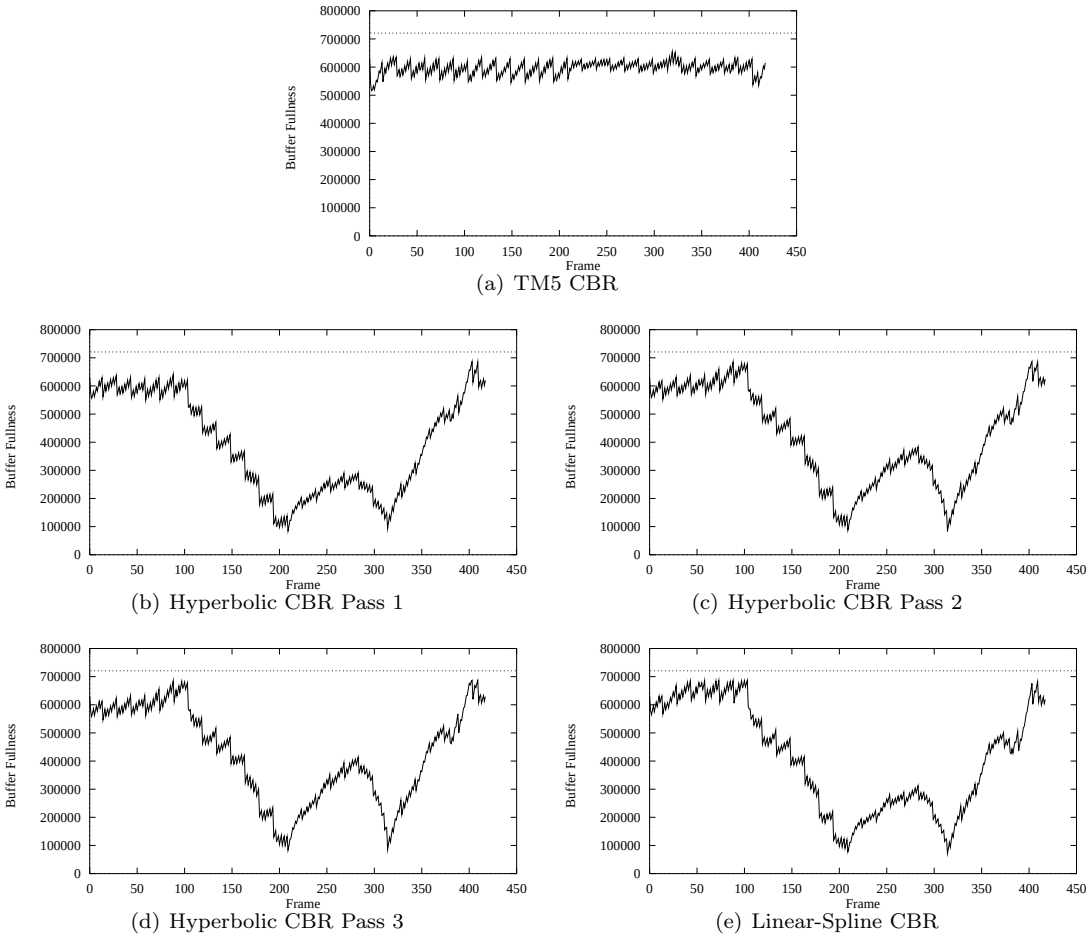
6.4 Buffer Guard Zones

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. This is illustrated in Figure 8. For VBR mode, the upper guard zone is not needed since buffer overflow is not a concern.

6.5 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 several short ($\approx$ 100 pictures) benchmark video sequences (`flower garden`, `football`, `mobile`, and `table tennis`) in SIF format and a longer (3000 pictures) promotional video clip courtesy of IBM Corporation.

[6]By nominal quantization scale, we mean the average measured macroblock quantization scale with perceptual quantization factored out.



(a) TM5 CBR

(b) Hyperbolic CBR Pass 1

(c) Hyperbolic CBR Pass 2

(d) Hyperbolic CBR Pass 3

(e) Linear-Spline CBR

Figure 9: Evolution of buffer fullness for CBR coders.

6.5.1 Initial Experiments

Initially, we used the short video clips to evaluate the bit-production models and rate control strategies. To simulate scene changes, we concatenated the four short video clips into a 418–picture video sequence in the following order: `flower garden`, `mobile`, `football`, `table tennis`.

We used the sample encoding parameters provided with the MPEG-2 Simulation Group software encoder. These parameters are listed in Table 1. For CBR mode, we specified a peak bit rate of 1.0 Mbit/sec. For VBR, we used an average bit rate of 1.0 Mbit/sec and a peak bit rate of 1.2 Mbit/sec. The VBV buffer size was set to 720,896 bits. For reference, we also ran the encoder with TM5 rate control using the sample parameters.

In the first set of simulations, we used the hyperbolic model and performed multiple encoding passes. The results of the encodings are presented in Table 2 and Figures 9 to 14. The table collects some summary statistics for the various coders. Figures 9 and 10 show the evolution of the buffer fullness. Figures 11 and 12 plot the targeted and observed nominal quantization scale Q. The targeted Q appear as piecewise constant segments, as dictated by theory.

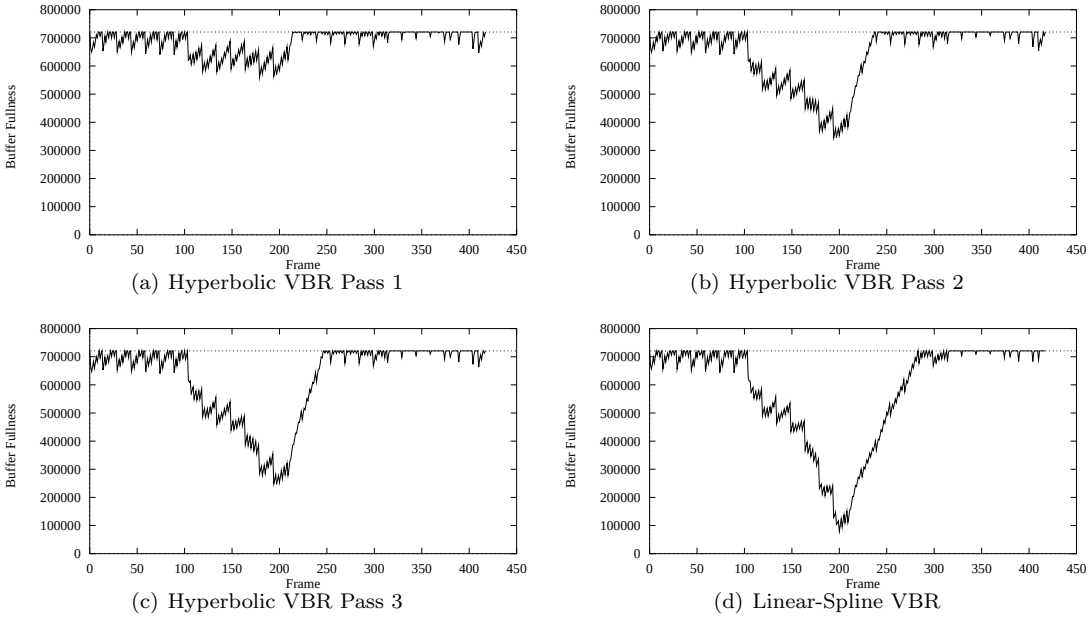
The initial pass of the Hyperbolic CBR and VBR coders used statistics gathered with the TM5 coder in order to determine the parameters of the bit-production models. Later passes of the Hyperbolic CBR (VBR) coder used statistics gathered from the previous pass. From the results in Table 2 and Figure 9, the

Value	Description
418	number of frames
1	number of first frame
15	N (# of frames in GOP)
3	M (I/P frame distance)
0	ISO/IEC 11172-2 stream
0	0:frame pictures, 1:field pictures
352	horizontal_size
240	vertical_size
2	aspect_ratio_information 1=square pel, 2=4:3, 3=16:9, 4=2.11:1
5	frame_rate_code 1=23.976, 2=24, 3=25, 4=29.97, 5=30 frames/sec
1000000.0	bit_rate (bits/sec)
44	vbv_buffer_size (in multiples of 16 kbit)
0	low_delay
0	constrained_parameters_flag
4	Profile ID: Simple = 5, Main = 4, SNR = 3, Spatial = 2, High = 1
8	Level ID: Low = 10, Main = 8, High 1440 = 6, High = 4
1	progressive_sequence
1	chroma_format: 1=4:2:0, 2=4:2:2, 3=4:4:4
2	video_format: 0=comp., 1=PAL, 2=NTSC, 3=SECAM, 4=MAC, 5=unspec.
5	color_primaries
5	transfer_characteristics
4	matrix_coefficients
352	display_horizontal_size
240	display_vertical_size
0	intra_dc_precision (0: 8 bit, 1: 9 bit, 2: 10 bit, 3: 11 bit
0	top_field_first
1 1 1	frame_pred_frame_dct (I P B)
0 0 0	concealment_motion_vectors (I P B)
1 1 1	q_scale_type (I P B)
1 0 0	intra_vlc_format (I P B)
0 0 0	alternate_scan (I P B)
0	repeat_first_field
1	progressive_frame
0	P distance between complete intra slice refresh
0	rate control: r (reaction parameter)
0	rate control: avg_act (initial average activity)
0	rate control: Xi (initial I frame global complexity measure)
0	rate control: Xp (initial P frame global complexity measure)
0	rate control: Xb (initial B frame global complexity measure)
0	rate control: d0i (initial I frame virtual buffer fullness)
0	rate control: d0p (initial P frame virtual buffer fullness)
0	rate control: d0b (initial B frame virtual buffer fullness)
2 2 11 11	P: forw_hor_f_code forw_vert_f_code search_width/height
1 1 3 3	B1: forw_hor_f_code forw_vert_f_code search_width/height
1 1 7 7	B1: back_hor_f_code back_vert_f_code search_width/height
1 1 7 7	B2: forw_hor_f_code forw_vert_f_code search_width/height
1 1 3 3	B2: back_hor_f_code back_vert_f_code search_width/height

Table 1: Parameters for MPEG-2 Simulation Group software encoder used to encode the SIF-formatted video clips.

Method	Average PSNR (dB)	Std. Dev. of PSNR	Average Nom. Q	Std. Dev. of Nom. Q	Maximum Nom. Q	Minimum Nom. Q
TM5 CBR	26.66	3.07	16.19	4.49	26.27	6.70
Hyperbolic CBR, Pass 1	26.48	2.67	16.49	3.50	22.79	7.91
Hyperbolic CBR, Pass 2	26.50	2.67	16.49	3.45	21.60	7.55
Hyperbolic CBR, Pass 3	26.48	2.67	16.54	3.47	22.01	7.22
Hyperbolic VBR, Pass 1	26.54	2.36	16.14	2.79	21.39	9.30
Hyperbolic VBR, Pass 2	26.53	2.12	16.19	2.18	19.82	9.49
Hyperbolic VBR, Pass 3	26.50	2.01	16.28	1.87	20.00	9.74
Linear-Spline CBR, Hybrid	26.73	2.68	15.80	3.36	19.48	8.29
Linear-Spline VBR, Hybrid	26.66	1.87	15.83	1.13	17.59	12.97

Table 2: Summary of initial coding experiments.



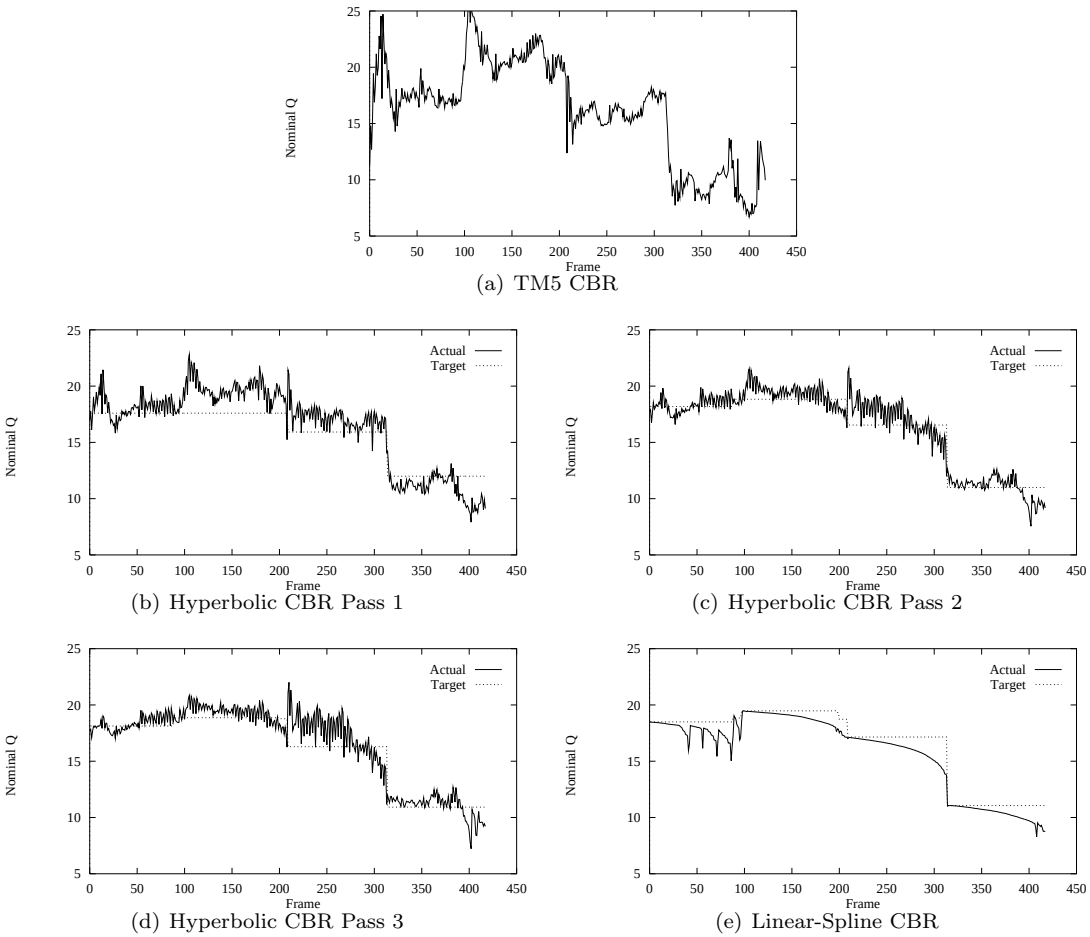
(a) Hyperbolic VBR Pass 1

(b) Hyperbolic VBR Pass 2

(c) Hyperbolic VBR Pass 3

(d) Linear-Spline VBR

Figure 10: Evolution of buffer fullness for VBR coders.



(a) TM5 CBR

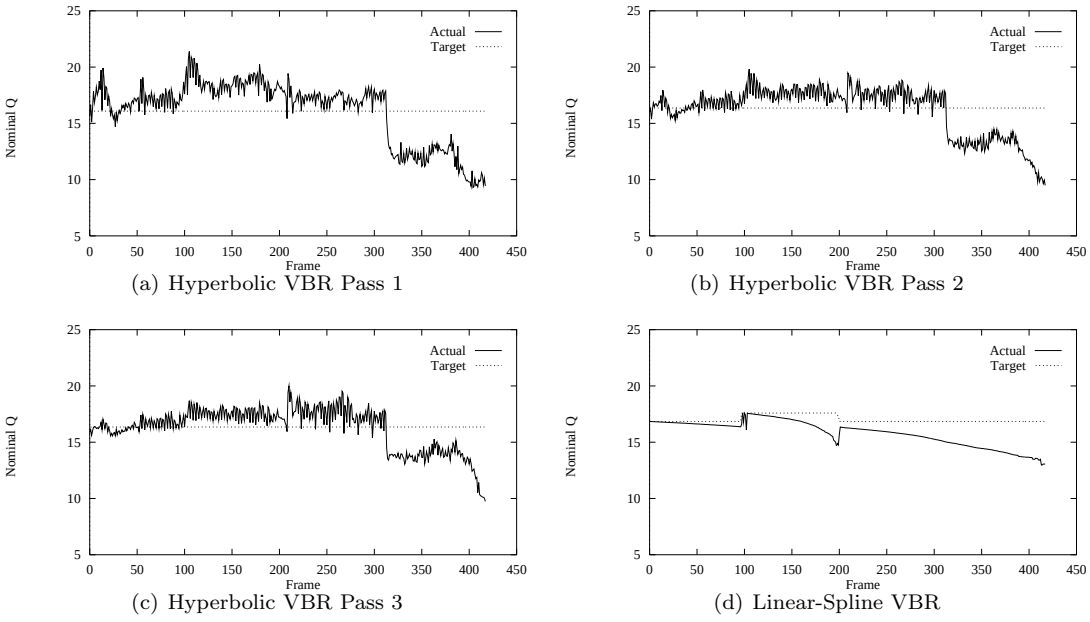
(b) Hyperbolic CBR Pass 1

(c) Hyperbolic CBR Pass 2

(d) Hyperbolic CBR Pass 3

(e) Linear-Spline CBR

Figure 11: Nominal quantization scale for CBR coders.



(a) Hyperbolic VBR Pass 1

(b) Hyperbolic VBR Pass 2

(c) Hyperbolic VBR Pass 3

(d) Linear-Spline VBR

Figure 12: Nominal quantization scale for VBR coders.

Hyperbolic CBR coder does not exhibit much change between passes. However, the Hyperbolic VBR coder does show reduction in the variances of PSNR and nominal Q and better usage of the VBV buffer with later passes.

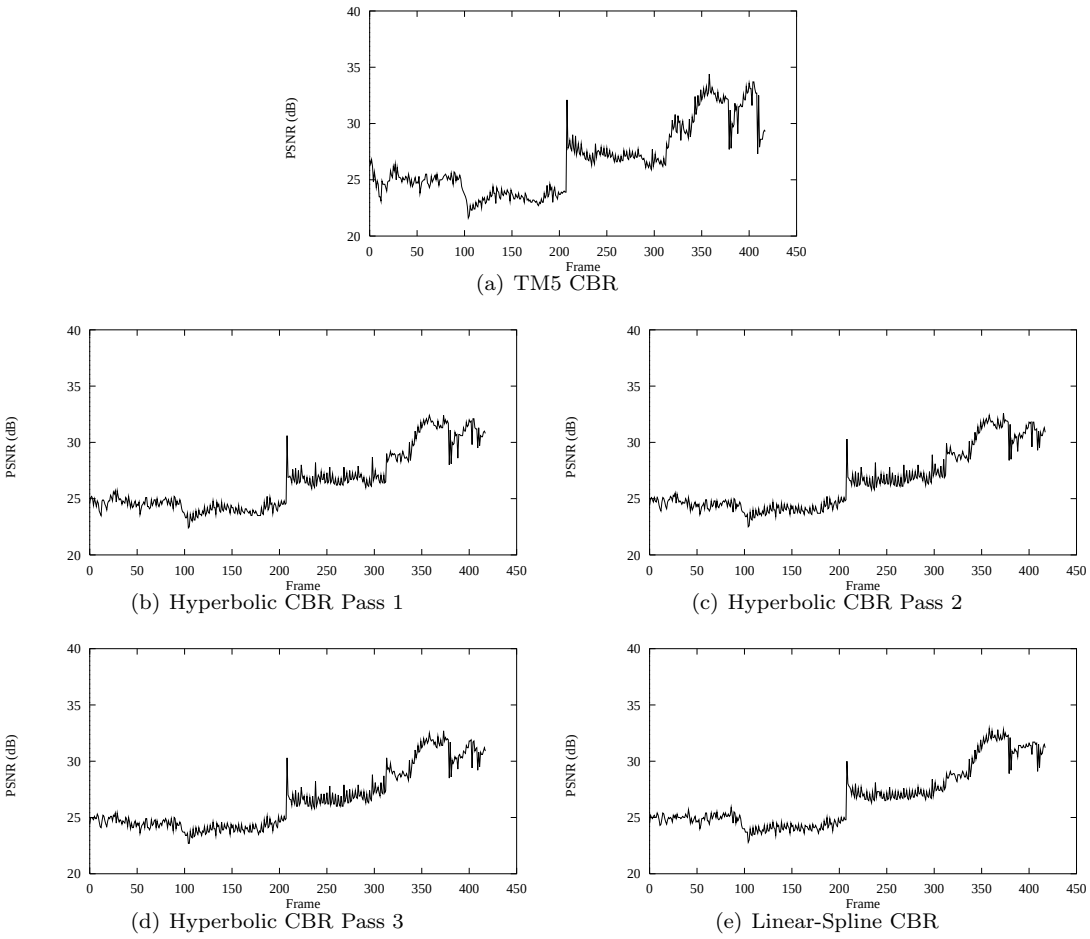
From Figure 9, we can see that the TM5 coder does not use much of the VBV buffer and keeps the buffer level fairly uniform. The lexicographic coders, on the other hand, make better use of the VBV buffer.

Comparing the hyperbolic model with closed-loop rate control on the one hand with the linear-spline model with hybrid rate-control on the other, we see that the latter outperforms the former in all aspects. It is noteworthy that hyperbolic model seems to underestimate the bit-production while the linear-spline model overestimates the bit-production. The result is that the actual nominal quantization scales used are higher than the target for the hyperbolic model and lower for the linear-spline model.

6.5.2 Coding a Longer Sequence

Since the video clips used in the initial experiments are short and we had to concatenate them to form somewhat artificial scene changes, we were not able to detect much perceptual difference between the different encodings. To assess the perceptual gains of lexicographically optimal bit allocation, we performed additional coding simulations using a longer video sequence with varied and dynamic content. The sequence consists of 3,660 frames of a commercial produced by the IBM Corporation to demonstrate its MPEG-2 encoding chipset. 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 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.

The video sequence is in NTSC CCIR-601 format. For the simulations, we coded the sequence in CBR mode at 3 Mbit/sec and in VBR mode at 3 Mbit/sec average and 4.5 Mbit/sec peak. The encoding parameters used are shown in Table 3. The VBV buffer size is 1,835,008 bits. We used linear-spline interpolation model in conjunction with the hybrid rate control strategy for the lexicographic coders.



(a) TM5 CBR

(b) Hyperbolic CBR Pass 1

(c) Hyperbolic CBR Pass 2

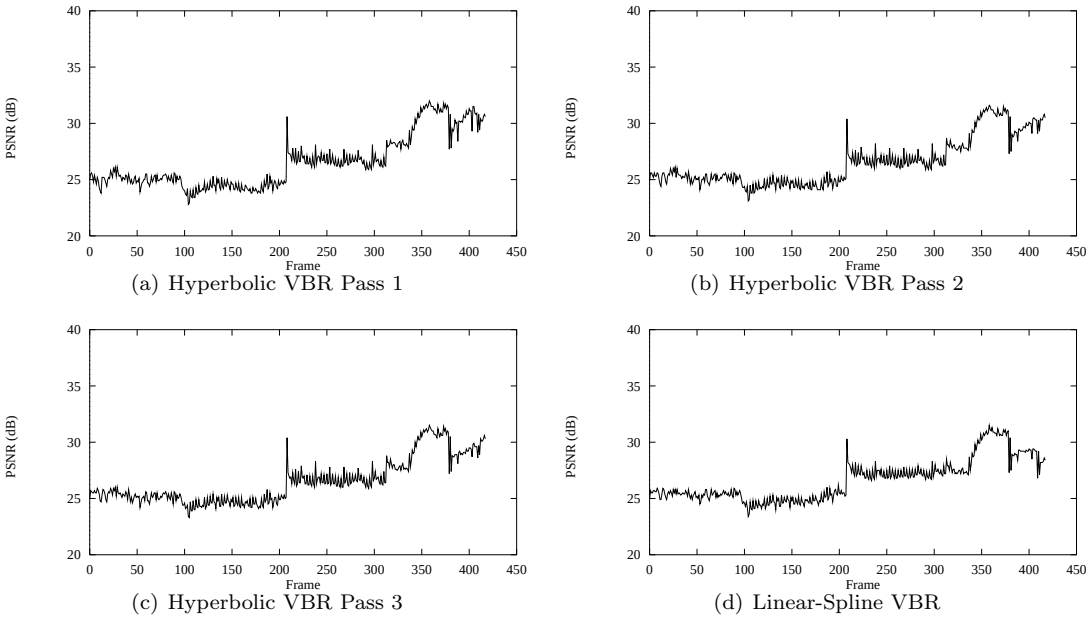
(d) Hyperbolic CBR Pass 3

(e) Linear-Spline CBR

Figure 13: PSNR for CBR coders.

Value	Description
3660	number of frames
0	number of first frame
15	N (# of frames in GOP)
3	M (I/P frame distance)
0	ISO/IEC 11172-2 stream
0	0:frame pictures, 1:field pictures
720	horizontal_size
480	vertical_size
2	aspect_ratio_information 1=square pel, 2=4:3, 3=16:9, 4=2.11:1
4	frame_rate_code 1=23.976, 2=24, 3=25, 4=29.97, 5=30 frames/sec
3000000.0	bit_rate (bits/sec)
112	vbv_buffer_size (in multiples of 16 kbit)
0	low_delay
0	constrained_parameters_flag
4	Profile ID: Simple = 5, Main = 4, SNR = 3, Spatial = 2, High = 1
8	Level ID: Low = 10, Main = 8, High 1440 = 6, High = 4
0	progressive_sequence
1	chroma_format: 1=4:2:0, 2=4:2:2, 3=4:4:4
2	video_format: 0=comp., 1=PAL, 2=NTSC, 3=SECAM, 4=MAC, 5=unspec.
5	color_primaries
5	transfer_characteristics
4	matrix_coefficients
720	display_horizontal_size
480	display_vertical_size
0	intra_dc_precision (0: 8 bit, 1: 9 bit, 2: 10 bit, 3: 11 bit
1	top_field_first
0 0 0	frame_pred_frame_dct (I P B)
0 0 0	concealment_motion_vectors (I P B)
1 1 1	q_scale_type (I P B)
1 0 0	intra_vlc_format (I P B)
0 0 0	alternate_scan (I P B)
0	repeat_first_field
0	progressive_frame
0	P distance between complete intra slice refresh
0	rate control: r (reaction parameter)
0	rate control: avg_act (initial average activity)
0	rate control: Xi (initial I frame global complexity measure)
0	rate control: Xp (initial P frame global complexity measure)
0	rate control: Xb (initial B frame global complexity measure)
0	rate control: d0i (initial I frame virtual buffer fullness)
0	rate control: d0p (initial P frame virtual buffer fullness)
0	rate control: d0b (initial B frame virtual buffer fullness)
4 4 63 63	P: forw_hor_f_code forw_vert_f_code search_width/height
2 2 15 15	B1: forw_hor_f_code forw_vert_f_code search_width/height
3 3 31 31	B1: back_hor_f_code back_vert_f_code search_width/height
3 3 31 31	B2: forw_hor_f_code forw_vert_f_code search_width/height
2 2 15 15	B2: back_hor_f_code back_vert_f_code search_width/height

Table 3: Parameters for MPEG-2 Simulation Group software encoder used to encode the IBM commercial.



(a) Hyperbolic VBR Pass 1

(b) Hyperbolic VBR Pass 2

(c) Hyperbolic VBR Pass 3

(d) Linear-Spline VBR

Figure 14: PSNR for VBR coders.

Method	Average PSNR (dB)	Std. Dev. of PSNR	Average Nom. Q	Std. Dev. of Nom. Q	Maximum Nom. Q	Minimum Nom. Q
TM5 CBR	33.30	4.93	14.62	11.25	61.34	1.95
Linear-Spline CBR, Hybrid	33.42	4.75	13.07	8.00	30.57	2.70
Linear-Spline VBR, Hybrid	33.06	2.58	11.84	2.05	17.68	7.95

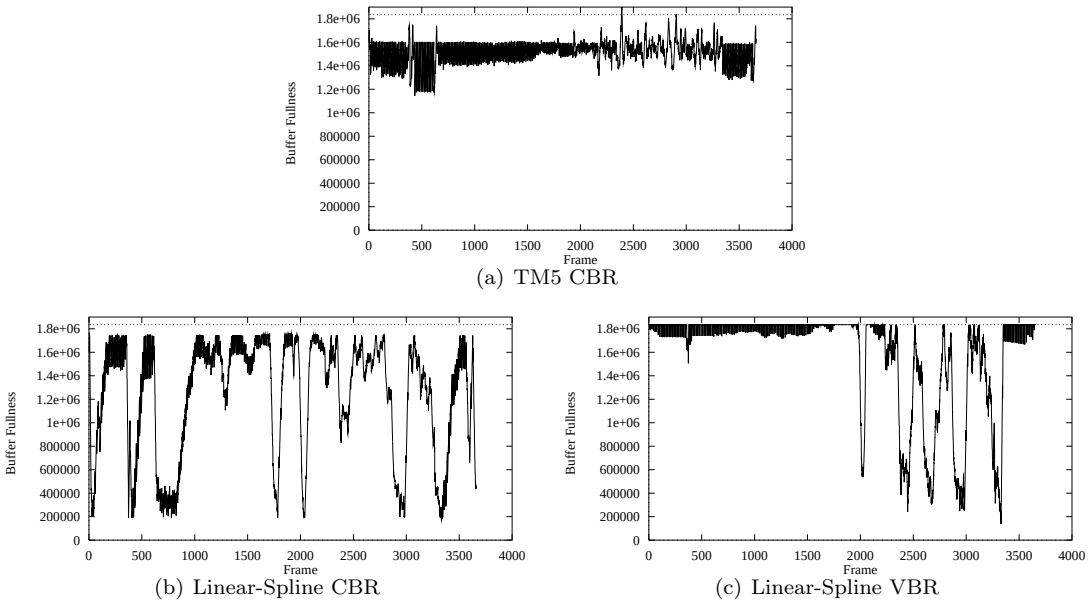
Table 4: Summary of coding simulations with IBM Commercial.

Some encoding statistics are listed in Table 4. The buffer fullness, nominal Q, and PSNR plots are shown in Figures 15, 16, and 17, respectively. The differences between the different coders are much more pronounced with these simulations than with the previous ones. 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 also 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.

6.6 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

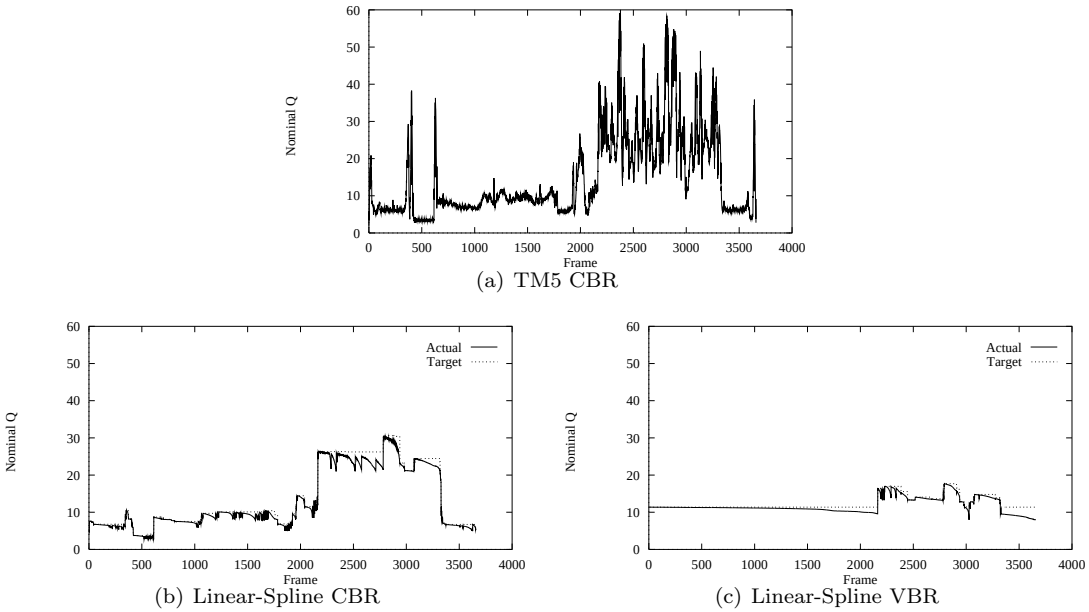


(a) TM5 CBR

(b) Linear-Spline CBR

(c) Linear-Spline VBR

Figure 15: Evolution of buffer fullness for coding IBM Commercial.

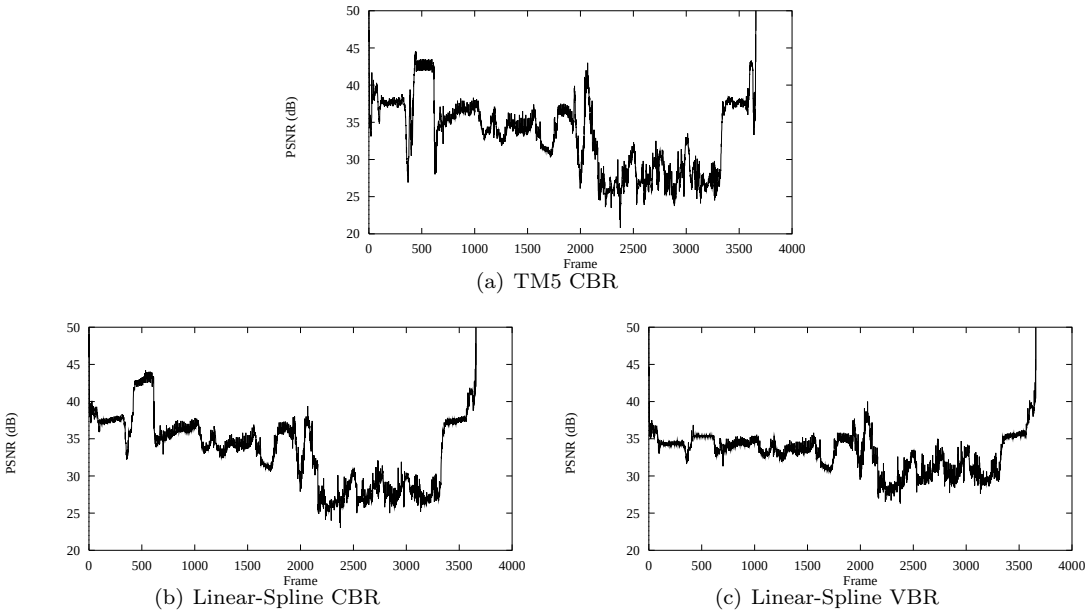


(a) TM5 CBR

(b) Linear-Spline CBR

(c) Linear-Spline VBR

Figure 16: Nominal quantization scale for coding IBM Commercial.



(a) TM5 CBR

(b) Linear-Spline CBR

(c) Linear-Spline VBR

Figure 17: PSNR for coding IBM Commercial.

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 $S < W$, the procedure is as follows:

1. Compute a bit allocation for the next W pictures not yet coded by computing a reverse dynamic programming table.
2. Code the next S pictures using hybrid rate control, using the dynamic programming table to recover from model errors.
3. Repeat Step 1.

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

Another approach similar to hybrid rate control method is to use the allocation computed from a given model and only recompute the allocation when the buffer fullness breach 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.

6.7 Related Work

In [8], a heuristic method is proposed 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 technique described in Section 6.3.2. 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 6.6. The authors also describe the heuristic of recomputing a allocation only when the buffer reaches predefined threshold levels.

In this section, we have considered a simple hyperbolic model and a linear-spline model of bit-production. We now review some previous work on bit modeling for video coding.

In [12], a complex 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 [35], 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 [31, 32] for use with a gradient-based rate-control algorithm [33, 34] that attempts to minimize MSE. The model takes into account the temporal dependencies introduced by predictive coding.

7 Extensions to the Lexicographic Framework

Thus far, we have provided a sound theoretical basis for lexicographic bit allocation and demonstrated that it works well in practice. In this section, we provide evidence that the framework is also flexible and general by showing how it can be readily applied to other application domains and with different sets of assumptions.

7.1 Applicability to Other Coding Domains

While the lexicographic bit allocation framework was originally motivated and formulated for MPEG video coding, it can be applied equally well in other lossy coding domains for which buffer-constrained bit allocation is a valid problem, and where a perceptual distortion measure needs to be equalized among coding units. Obvious examples include lossy image coding (such as specified by the JPEG standard [36]), where the coding unit would logically be a block of pixels, and audio coding, where a coding unit might correspond to half a second of sampled sound.

In the original formulation, we identified a perceptually-adjusted quantization scale as the object of optimization. In general, any signal that can be controlled to affect the bit production of the encoder can be used.

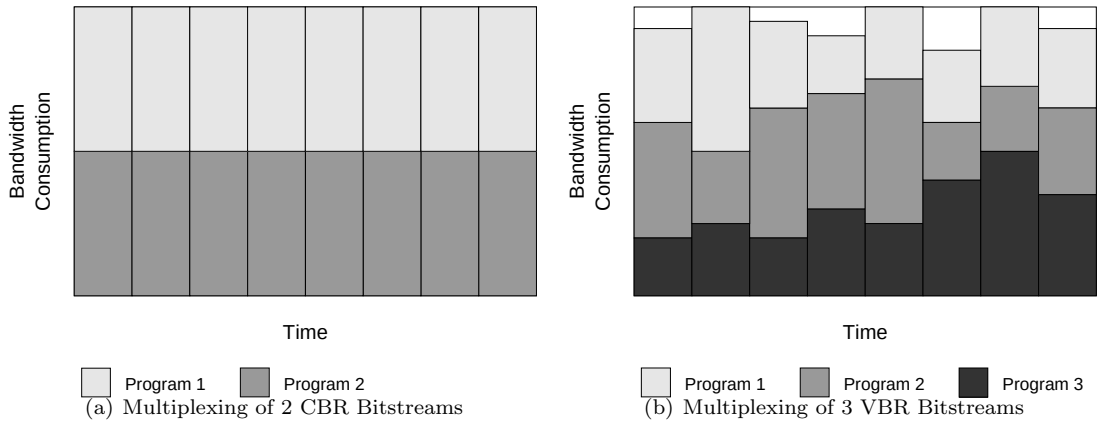
7.2 Multiplexing VBR Streams over a CBR Channel

7.2.1 Introduction

There are many scenarios where multiple compressed video streams are to be transmitted through a common channel. Two obvious examples are networked video and digital video broadcasting. In these types of applications, the transmission channel is typically bandwidth-limited. With the available bandwidth, we would like to provide as much video programming as possible without having to sacrifice quality.

An often-cited motivation for VBR video encoding is that VBR encoding could potentially allow for the simultaneous transmission of more video streams over a common channel than CBR coding at the same level of quality. The main reasoning is provided through a concept called *statistical multiplexing*. Statistical multiplexing is based on the observation that the bit rate of constant-quality video is highly variable from frame to frame. In order to achieve image quality that is *not less* lexicographically than that of a constant-quality VBR encoding, a CBR encoding would require a bit rate that would correspond to the peak rate of the VBR encoding. Since a VBR encoding typically requires the peak rate for only a small percentage of time, it uses less bandwidth on average than a comparable CBR encoding. Furthermore, assuming that the VBR streams have independent bit-rate characteristics, we can transmit more VBR streams than CBR streams over a common channel with a low probability that the combined instantaneous bit rate would exceed the channel rate.

As an example, consider a channel with a bandwidth of 100 Mbit/sec. Suppose that for a desired level of quality, a peak rate of 10 Mbit/sec is required for coding a suite of video programming. Using CBR encoding, up to 10 sequences can be transmitted through the channel simultaneously. Since the peak rate is required only a small percentage of the time, suppose that the actual average rate is only 5 Mbit/sec. Then using VBR encoding with a peak rate of 10 Mbit/sec and average rate of 5 Mbit/sec, we can *potentially* transmit 20 simultaneous sequences. This would correspond to a *statistical multiplexing gain* of 2.



(a) Multiplexing of 2 CBR Bitstreams (b) Multiplexing of 3 VBR Bitstreams

Figure 18: Example of how three VBR bitstreams can be multiplexed into the same channel as two CBR bitstreams, for a statistical multiplexing gain of 1.5.

In order to transmit the 20 VBR sequences simultaneously, however, the instantaneous bit rate for the 20 sequences must not exceed the channel capacity for an extended period of time, which is determined by the amount of buffering present. Assuming that the bit rates of the different video streams are uncorrelated in time, there is a low probability that the channel capacity would be exceeded in any time interval. Quantifying and minimizing this probability are central themes of research.

The advantage of VBR encoding over CBR is illustrated through a simple example in Figure 18. In this example, three VBR encodings are shown multiplexed using at most the same bandwidth required by a CBR encoding of only two of the sources.

In this remainder of this section, we will show how our basic lexicographic bit allocation framework can be readily extended to handle the multiplexing of multiple VBR bitstreams over a CBR channel. However, in contrast to typical statistical multiplexing techniques, as exemplified in Figure 18, our method allocates bits to the VBR bitstreams in a *deterministic* manner, making full use of all the available channel bandwidth.

In related work, a buffered rate control scheme for multiplexing VBR sources onto a CBR channel is described in [37]. This work is based on the rate-distortion framework of [1] and [38] and uses a multiplexing model very similar to the one we are about to present. As described in the paper, the basic allocation unit is taken to be a GOP.

7.2.2 Multiplexing Model

We first elaborate a model for multiplexing multiple VBR bitstreams onto a CBR channel. Since our bit allocation framework is deterministic and uses lookahead, we assume that complete statistics of the multiple video sources are available to the bit allocation algorithm. This requirement can be met by providing a centralized encoder for the multiple sources, as depicted in Figure 19. In the figure, M video sources enter a encoder/multiplexer that produces a single multiplexed stream for transport over a CBR channel. On the receiving end, a demultiplexer/decoder performs demultiplexing and decoding to reproduce the M video sequences. This multiplexing model is similar to that proposed in [28].

This model is applicable to applications such as a video server where the video sequences to be multiplexed are known in advanced. An especially noteworthy case is that of near-video-on-demand (NVOD), where a single sequence is to be transmitted simultaneously with different starting times. For example, 20 copies of a 2-hour long movie can be multiplexed so that the viewing of the movie can begin every six minutes.

The encoder/multiplexer block is expanded in Figure 20. As shown, the input video sources are encoded individually and time-division multiplexed and stored in a buffer before being output to the channel at a constant bit rate. The encoders are synchronized so that they output the encoding of a picture at the same

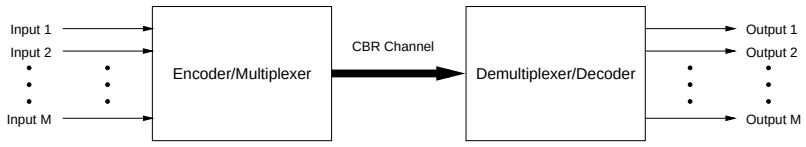


Figure 19: System for transmitting multiple sequences over a single channel.

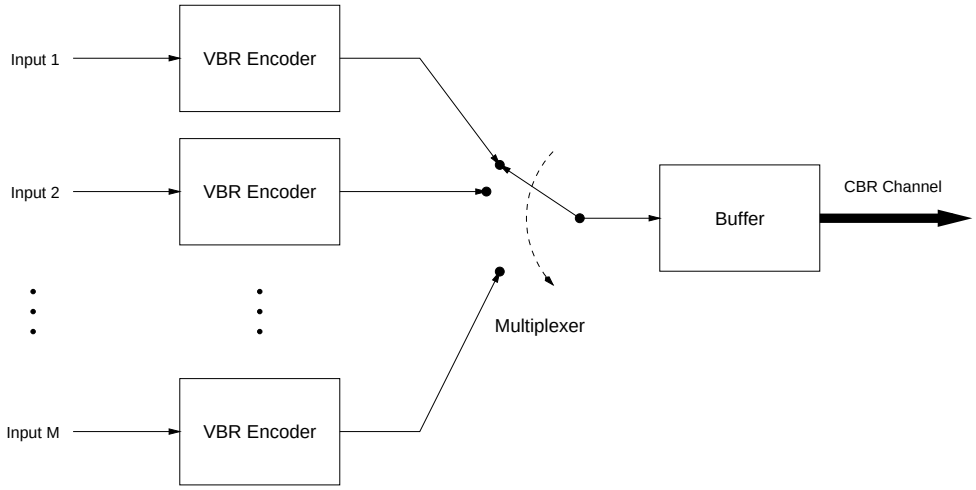


Figure 20: Block diagram of encoder/multiplexer.

time every T seconds. The multiplexer then concatenates the multiple encodings in order as shown in Figure 21.

The demultiplexer/decoder block is expanded in Figure 22. The demultiplexer/decoder mirrors the operation of the encoder/multiplexer. Incoming bits from the channel are stored in a decoding buffer. Every T seconds, the demultiplexer instantaneously removes from the buffer all bits needed to decode the next picture of all sequences and routes the bitstreams to the appropriate decoders, which then output the reconstructed video.

The multiplexing model described above resembles the operation of the single-stream encoder and decoder system implied by the MPEG Video Buffering Verifier. If we view the different input sources as providing "slices" of the same picture, the resemblance would be very close indeed. This construction is intentional and allows us to apply the lexicographic framework to allocate bits optimally to the multiple VBR bitstreams.

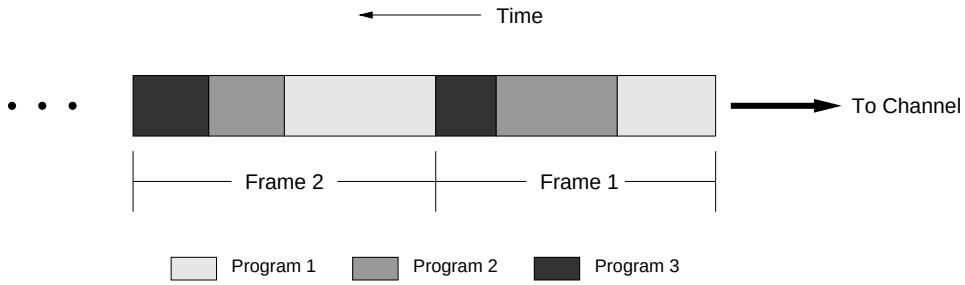


Figure 21: Operation of multiplexer.

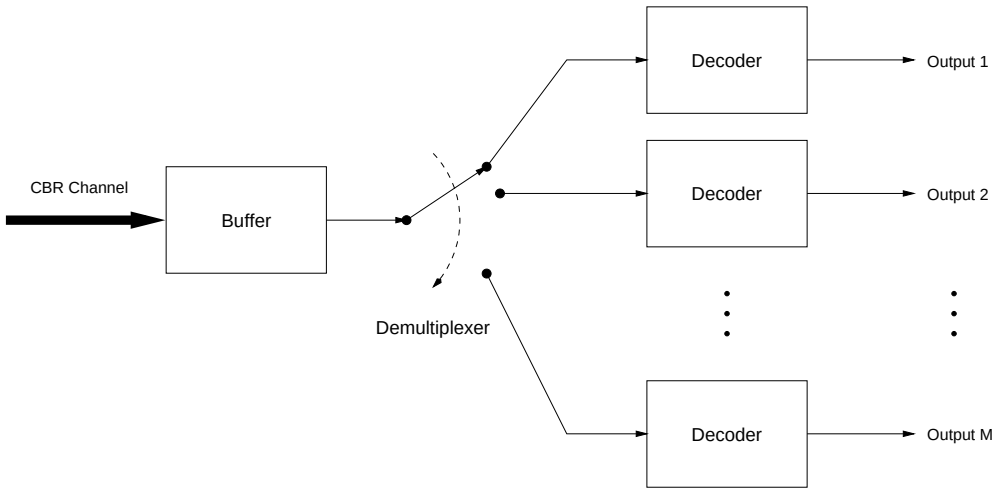


Figure 22: Block diagram of demultiplexer/decoder.

7.2.3 Lexicographic Criterion

Before we can apply our lexicographic bit allocation framework, we need to define an optimality criterion. Since there are multiple sources, a lexicographical criterion based on the encoding of a single video sequence is certainly not appropriate. We need to consider the quality of all the sequences. A simple way to do this is to consider the concatenation of all the sequences and define a lexicographic criterion on the concatenated video stream. By doing this, we are putting equal weight to each picture of each video sequence. We can also consider extending the lexicographic criterion to compare vectors of length M instead of scalar quantities. However, as we will see, this is equivalent to just considering the concatenation of the M sequences.

7.2.4 Equivalence to CBR Bit Allocation

In this section, we show in the multiplexing model put forth above that the problem of optimal bit allocation for multiple VBR streams reduces to a CBR bit allocation problem for a single stream.

The multiplexing model is intentionally set up to follow the operation of the MPEG Video Buffering Verifier (VBV). We can view the buffer, the demultiplexer, and the bank of M decoders in Figure 22 as comprising a single VBV. The lexicographic framework of Section 3 can then be applied. Since the transmission channel operates at a constant bit rate, the CBR constraints of Section 3.3.1 would apply.

For display interval i, we need to consider the quantization scales used to code picture i of each of the M video sequences. Given a fixed bit budget for coding picture i of each video sequence, it is easy to show that the same quantization scale must be used to code picture i of all sequences to achieve lexicographic optimality; if the quantization scales differ, we can always shift bits around to reduce the highest quantization scale by increasing a lower quantization scale. This result also holds if we formulate the lexicographic criterion using vectors of quantization scales.

By using a combined bit production model that is the sum of the bit production models for the individual sequences, we can then allocate bits jointly to the sequences using the CBR Algorithm of Section 3.3.1.

While the above technique guarantees that the buffer in Figure 22 does not overflow or underflow, it should be noted that doing so *does not* guarantee MPEG VBV compliance for the individual sequences, except when the individual VBV buffers are at least the size of the buffer in Figure 22. A multiplexing model that explicitly includes individual decoder buffers is certainly possible. However, analysis of this situation is not as straightforward as the above model and remains an open problem.

7.3 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 5, 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 6, 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 method proposed in Section 6.

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 [1]. 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.

7.3.1 Dynamic Programming

The dynamic programming algorithm described in [1] is based on the Viterbi Algorithm described in [3] 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 [1] requires $O(MBN)$ time to compute an optimal bit allocation.

7.3.2 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.

References

[1] 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.

[2] K. Ramchandran, A. Ortega, and M. Vetterli, "Bit allocation for dependent quantization with applications to multiresolution and MPEG video coders", *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 533–545, Sept. 1994.

[3] 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.

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

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

[6] K. M. Uz, J. M. Shapiro, and M. Czigler, "Optimal bit allocation in the presence of quantizer feedback", in *Proceedings ICASSP'93*, 1993, vol. 5, pp. 385–388.

[7] 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.

[8] 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.

[9] K. Ramchandran, A. Ortega, and M. Vetterli, "Bit allocation for dependent quantization with applications to MPEG video coders", in *Proceedings ICASSP'93*, 1993, vol. 5, pp. 381–384.

[10] C.-Y. Hsu, A. Ortega, and A. R. Reibman, "Joint selection of source and channel rate for VBR video transmission under ATM policing constraints", 1997, To appear.

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

[12] 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.

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

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

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

[24] 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.

[25] 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.

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

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

[28] 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.

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

[30] 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.

[31] L.-J. Lin, A. Ortega, and C.-C. J. Kuo, "Rate control using spline-interpolated R-D characteristics", in *Proceedings VCIP'96*, 1996.

[32] L.-J. Lin, A. Ortega, and C.-C. J. Kuo, "Cubic spline approximation of rate and distortion functions for MPEG video", in *Proceedings SPIE 1996 Digital Video Compression Conference*, San Jose, CA, Jan. 1996.

[33] L.-J. Lin, A. Ortega, and C.-C. J. Kuo, "Gradient-based buffer control techniques for MPEG", in *Proceedings VCIP'95*, Taipei, Taiwan, May 1995.

[34] 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.

[35] 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.

[36] W.B. Pennebaker and J. L. Mitchell, *JPEG—Still Image Data Compression Standard*, Van Nostrand Reinhold, New York, 1993.

[37] D. Park and K. Kim, "Buffered rate-distortion control of MPEG compressed video channel for DBS applications", in *Proceedings IEEE Internation Conference on Communications*, 1995, vol. 3, pp. 1751–1755.

[38] J. Choi and D. Park, "A stable feedback control of the buffer state using the controlled lagrange multiplier method", *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 546–557, Sept. 1994.