Optimal bit allocation for fine-grained scalable video sequences in distributed streaming environments

ChengHsin Hsu and Mohamed Hefeeda

Network Systems Laboratory
School of Computing Science
Simon Fraser University
Surrey, BC, Canada

ABSTRACT

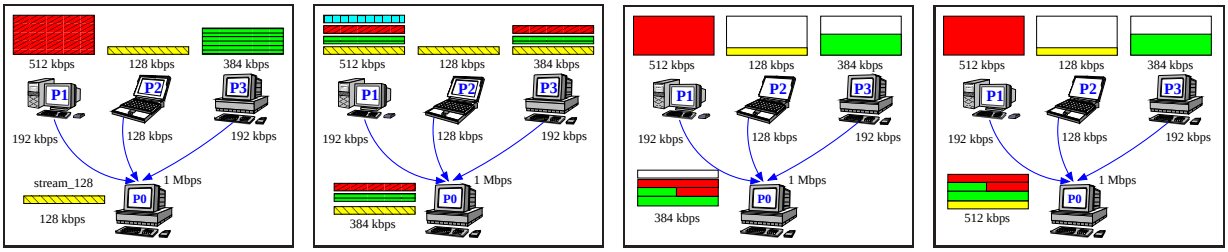
We present optimal schemes for allocating bits of fine-grained scalable video sequences among multiple senders streaming to a single receiver. This allocation problem is critical in optimizing the perceived quality in peer-to-peer and distributed multi-server streaming environments. Senders in such environments are heterogeneous in their outgoing bandwidth and they hold different portions of the video stream. We formulate the allocation problem as an optimization problem, which is nonlinear in general. We use rate-distortion models in the formulation to achieve the minimum distortion in the rendered video, constrained by the outgoing bandwidth of senders, availability of video data at senders, and incoming bandwidth of receiver. We show how the adopted rate-distortion models transform the nonlinear problem to an integer linear programming (ILP) problem. We then design a simple rounding scheme that transforms the ILP problem to a linear programming (LP) one, which can be solved efficiently using common optimization techniques such as the Simplex method. We prove that our rounding scheme always produces a feasible solution, and the solution is within a negligible margin from the optimal solution. We also propose a new algorithm (FGSAssign) for the allocation problem that runs in $O(n \log n)$ steps, where n is the number of senders. We prove that FGSAssign is optimal. Because of its short running time, FGSAssign can be used in real time during the streaming session. Our experimental study validates our analytical analysis and shows the effectiveness of our allocation algorithm in improving the video quality.

Keywords: Fine-grained scalable streaming, FGS, rate-distortion optimized streaming, video streaming, peer-to-peer streaming, distributed streaming

1. INTRODUCTION

Video streaming over the Internet is increasingly getting very popular. In this paper, we consider video streaming systems in which a streaming session has multiple senders and a single receiver. Multiple senders may be required in peer-to-peer streaming environments,[1–3] because of the limited capacity and unreliability of peers. Multiple senders are also desired in distributed streaming systems[4] to achieve disjoint network path streaming and hence better quality. We consider the general case when senders have heterogeneous outgoing bandwidth, and may store different portions of the requested stream. Our problem is to optimally allocate to each potential sender a transmission rate and range of bits to transmit such that the best video quality is achieved at the receiver. We illustrate the importance of the bit allocation problem using the simple example shown in Figure 1. There are three senders P1, P2, and P3, and one receiver P0. P1, P2, and P3 have outgoing bandwidth of 192, 128, and 192 Kbps, respectively, and P0 has incoming bandwidth of 1 Mbps. Furthermore, senders are assumed to have downloaded different portions of the stream in the past.

Figure 1 compares various possible solutions for the allocation problem, and the quality of the received stream in each case. In Figure 1(a), a nonscalable allocation scheme is considered, which assumes that the video stream is encoded using a nonscalable encoder. In nonscalable encoding, the stream has to be downloaded in its entirety, otherwise it is not decodable. In this case, senders are considered to have different *versions* of the same stream. Therefore, under the nonscalable allocation scheme, the receiver can only get the 128 Kbps stream version. Notice that collaboration among senders is not possible, because the versions they have are encoded differently. Nonscalable encoding is actually not uncommon in current Internet streaming systems: Many streaming servers post the same video clip in different rates to accommodate users with heterogeneous bandwidth. To cope with the inflexibility of nonscalable encoding, some streaming systems encode a stream into multiple layers. Typically, there are a few number of layers, because of the layering overhead and the complexity of the coding/decoding processes. In layered scalable coding, partial layers are not decodable. Figure



(a) Nonscalable (b) Layered scalable (c) FGS scalable (d) Optimal FGS scalable

Figure 1. A multi-sender streaming session using different allocation schemes.

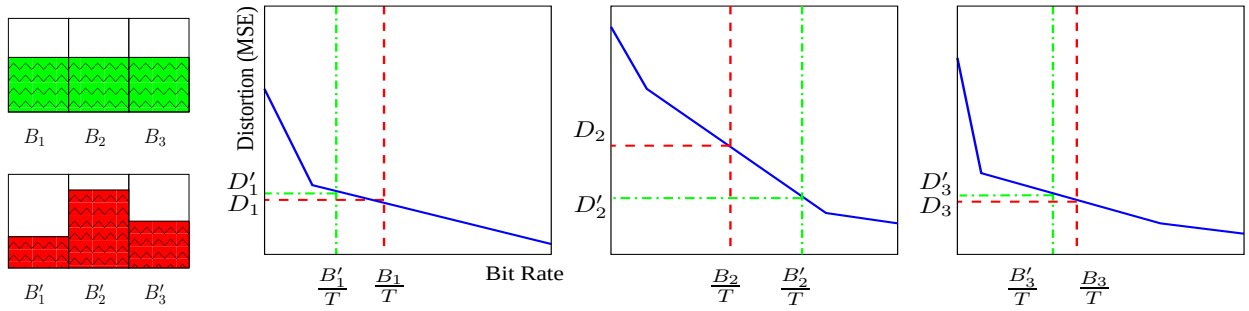
1(b) shows a layered scalable allocation scheme, where the stream is encoded into four layers, each has a bit rate of 128 Kbps. In this case, the receiver can get up to three layers, one from each sender. Thus the received stream will be of rate 384 Kbps. Due to the coarse-grain nature of the layers, this allocation scheme cannot leverage the remaining 64 Kbps bandwidth at each of P1 and P3.

The fine-granularity scalability (FGS) encoding adds significant flexibility to the layered encoding: FGS-encoded streams provide bit-level scalability, which means that bit streams can be truncated at any bit location. In addition, FGS-encoded streams have the property that a low-quality stream is always a prefix of a higher-quality one. The FGS flexibility, however, complicates the problem of allocating bits to senders, because of the finer resolution and the too many allocation possibilities that should be considered to select the optimal allocation. For instance, a possible FGS allocation is shown in Figure 1(c), where the received stream has a bit rate of 384 Kbps. This scheme started by allocating the first 192 Kbps of the stream to P3. Thus it cannot use P2, because the stream at P2 is a prefix of what would be transmitted by P3. The only option left for this allocation scheme is to allocate the second 192 Kbps of the stream to P1. A more careful—actually optimal—allocation is shown in Figure 1(d), where the receiver gets a much better quality stream of 512 Kbps bit rate.

To summarize, this example indicates that significant quality improvement could be achieved by adopting the fine-grained scalable encoding in streaming systems with multiple heterogeneous senders. It also highlights the importance of the optimal allocation of bits among senders.

The bit allocation problem described above can be solved on a frame-by-frame basis: For every frame, we optimally allocate the bits of that frame to senders. The optimal allocation ensures that the maximum number of bits in every frame is transmitted from senders. More bits of the same frame yield less reconstruction distortion, and hence better playout quality. The bit allocation problem could also be solved for a block of multiple frames at once. The rationale is that we may have an opportunity to further enhance quality by considering the relative importance of bits in frames belonging to the same block.

To illustrate the multiple-frame bit allocation problem, consider a block of three frames, as shown in Figure 2(a). Assume that the optimal solution of the single-frame allocation problem determined that senders can transmit up to $B_1 = B_2 = B_3$ bits for frames 1, 2, and 3. Further, suppose that the frames have the rate-distortion (R-D) curves shown in Figure 2(b). R-D curves map a given bit rate to the corresponding distortion level (detailed description of R-D curves is given in Section 3). Notice that the R-D curves may differ from one frame to another, because they depend on the temporal and spatial complexities of the frame. Figure 2(b) indicates that transmitting B_1, B_2, B_3 bits results in distortion levels D_1, D_2, D_3 for frames 1, 2, and 3, respectively. Notice that D_2 is much larger than D_1 and D_3, which implies large fluctuation in the playout quality of the consecutive frames. Quality fluctuations have a negative impact on the user-perceived quality. Now, consider the same bit budget for the three-frame block, but distributed differently among the frames, as shown in the lower part of Figure 2(a). This slight change in bit distribution results in quality improvement in two ways: (i) smaller total distortion in the block; and, more importantly, (ii) much smaller quality fluctuation in successive frames. This is shown in Figure 2(b), where the new distortion levels are denoted by D'_1, D'_2, D'_3. Notice that the decrease in bit rates allocated to frames 1 and 2 result in a minor increase in their distortion levels, while the distortion of frame 2 decreases substantially due to the extra bit rate allocated to it. In summary, solving the bit allocation problem at the block level optimizes the playout quality by getting the maximum number of bits from senders, and by carefully distributing these bits among frames using rate-distortion curves.



(a) Bits allocated to each frame in a block

(b) Rate-distortion curves for three consecutive frames in a block

Figure 2. The importance of solving the bit allocation problem at the block level.

In this paper, we formulate and optimally solve the single-frame bit allocation problem, and we experimentally show the effectiveness of our solutions in improving the video playout quality. In particular, we make the following contributions. First, we formulate the single-frame allocation problem as an optimization problem with an objective function to minimize the distortion. By using the piece-wise linear rate-distortion model, we transform the general (nonlinear) optimization problem into an integer linear programming (ILP) problem. We design a simple rounding scheme that transforms the ILP problem into a linear programming (LP) one, which could be solved efficiently using common optimization techniques such as the Simplex method. We prove that our rounding scheme always produces a feasible solution, and the solution is within a negligible margin from the optimal one. Second, we propose a new algorithm (FGSAssign) for the single-frame allocation problem that runs in $O(n \log n)$ steps, where n is the number of senders. We prove that FGSAssign is optimal. We solve the multiple-frame allocation problems in the extended version of this paper.[5]

The rest of this paper is organized as follows. We discuss related work in Section 2. Section 3 presents and validates the rate-distortion model adopted in this paper. In Section 4, we formulate the single-frame allocation problems, and we present our rounding scheme. In Section 5, we present the new FGSAssign algorithm, and we prove its optimality. We evaluate our algorithm and compare it against the optimal one in Section 6. Section 7 concludes the paper.

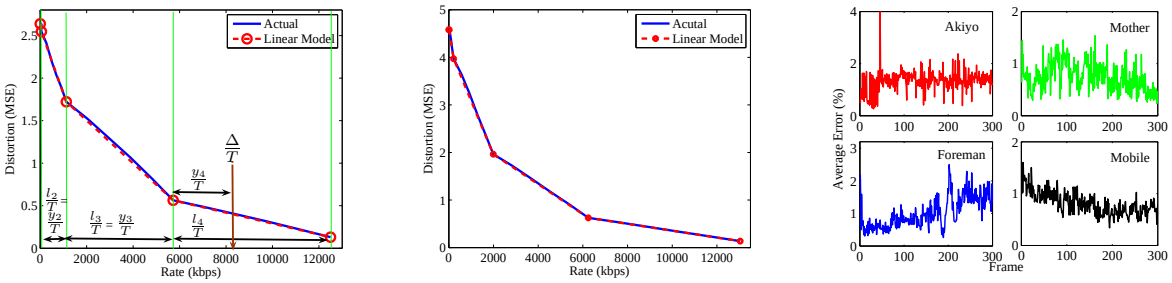
2. RELATED WORK

Distributed and peer-to-peer streaming has recently received significant research attention. For example, the distributed video streaming framework[4] shows the feasibility and benefits of streaming from multiple servers to a single receiver. The receiver uses a rate allocation algorithm to determine the sending rate for each server to minimize the total packet loss. A tomography-based sender selection protocol is proposed in[1] to optimize quality at the receiver. Both work do not consider scalable-coded streams. Layered-scalable streams are considered in[3] and[2] to cope with the bandwidth heterogeneity. In,[3] the multi-sender streaming problem is formulated to maximize the streaming quality of all peers and minimize the load on the originating media distributor. The authors of[2] design practical algorithms to adapt to bandwidth dynamics. Different from our work, these two studies employ coarse-grained scalability, and they are not R-D optimized.

Su and Wang consider a minimization problem for the transmission time of FGS-encoded images in.[6] They formulate the problem as a nonlinear program, and then they transform it to a series of LP subproblems. Each LP subproblem determines a peer allocation to maximize the number of received bits in a given delay bound. Unlike our work, the work in[6] does not consider the characteristics of the R-D curves of video sequences. A framework to solve the problem of streaming interdependent packetized video data units over lossy networks in R-D optimized fashion is proposed in.[7] This framework has been generalized by many researchers from various perspectives. For instance,[8] and[9] consider the multiple server video streaming problem in R-D optimized way. However, they concentrate on nonscalable and layered coded streams, and do not explicitly specify any R-D models.

3. RATE-DISTORTION MODELS

Rate-distortion (R-D) models are functions that map bit rates to expected distortion—and hence perceived quality—level. Since we are interested in optimizing quality at various rates, we need accurate R-D models. Several analytic R-D models



(a) Akiyo sequence, frame 150 (b) Mother sequence, frame 110 (c) The average error after aggregations

Figure 3. The accuracy of the linear R-D model: (a), (b) randomly chosen frames from different sequences, and (c) the average error between curves estimated by the (modified) linear model and the actual R-D curves.

for FGS-encoded sequences have been proposed, e.g., the square-root model,[10] the logarithm model,[11] and the Generalized Gaussian model.[12] The accuracy and complexity of various FGS R-D models are studied in.[13] In each one of these analytic models, the distortion is related to the rate with a *nonlinear* function. Nonlinear R-D functions make our bit allocation problem (described in Section 4) a nonlinear optimization problem, which is hard (if at all feasible) to solve.

Another approach for obtaining R-D models is by empirically measuring the distortion at various bit rates. Since the range of possible bit rates for the enhancement layer is typically large (order of Mbps), too many sample bit rates are needed to obtain accurate R-D models. This requires decoding the video sequence many times, which is computationally expensive [14, page 302]. A third approach for constructing R-D models is to empirically measure distortion only at a few carefully chosen bit rates and interpolate the R-D curve between these points based on some inherent characteristics of the enhancement layer. The piece-wise linear R-D model,[15] which we adopt in this paper, falls into this category.

In the following subsections, we first present the piece-wise linear R-D model and how we extract its parameters from video sequences. Then, we experimentally show its accuracy. Finally, we discuss important properties of this model which we will employ in the formulation and solution of our bit allocation problem.

3.1. The Piece-wise Linear R-D Model

The key idea of the piece-wise linear model[15] (we refer to it simply as the linear model) is that within each bitplane the R-D curve can be approximated by a line segment. Line segments of different bitplanes have different slopes. Figure 3(a) shows an example. The problem of finding the R-D function is now reduced to measuring the distortion at bit rates that correspond to bitplane boundaries (up to 8 samples in most cases), and computing the slopes of the different line segments.

Building and storing the linear R-D model for a given sequence is quite efficient. We first identify the bitplane boundaries from the header inserted in each bitplane during the FGS encoding process. We compute the size of each bitplane h, and denote it by l_h. If there are z bitplanes, we decode the sequence $z+1$ times but in each time we *truncate* the enhancement layer at a different bitplane boundary. After decoding we compute the distortion—in mean-square error (MSE)—between the original and reconstructed sequences. Then we compute the slope g_h for each bitplane h. The R-D model parameters are extracted only once and stored in a meta file. We need to store only the size of each bitplane and the slope of the line segment in that bitplane. The meta file adds a negligible storage overhead, up to 64 bytes per frame. Frames sizes are usually in order of tens of kilo bytes.

3.2. Accuracy of the Linear R-D Model

Since the accuracy of the adopted R-D model is crucial to our problem, we validate the accuracy of the linear R-D model. To do so, we compare the accuracy of the linear model against the actual R-D curves, which are obtained as follows. We select six equally-spaced sampling rates on each bitplane. Then, we compute the actual distortion by decoding and comparing the original and reconstructed sequences at each sampling rate. We carry out the comparison over several video sequences of different temporal and spatial complexities.

For visual validation, we randomly choose two frames and plot the actual R-D curves and the ones estimated by the linear model. Figure 3(a) and 3(b) illustrate that the linear model approximates the actual R-D curves quite well. For more

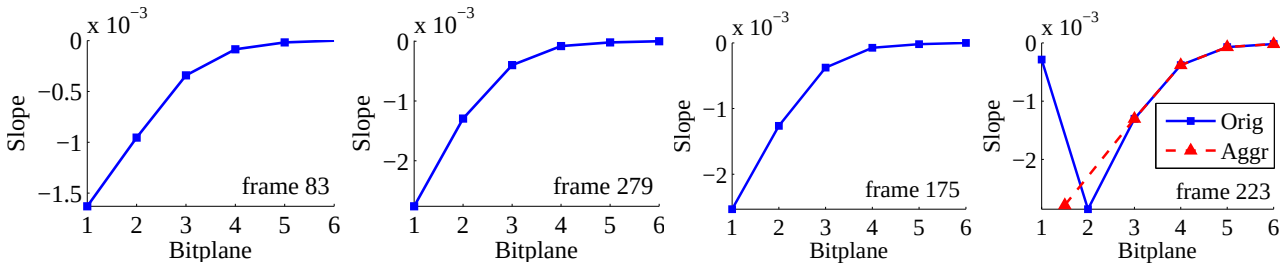


Figure 4. The monotonicity property of the linear R-D model. The figure shows the slope parameters g of two frames from Foreman (left) and two frames from Mobile (right) sequences. Note that, only the rightmost frame requires aggregation.

rigorous validation, we compute the absolute error between the actual and linear model curves at all sampling rates. We compute the average error per frame, and we repeat that for all frames in the considered sequences. The results (plots are given in[5]) confirm the accuracy of the linear R-D model: the average error is less than 2% in almost all cases.

3.3. Properties of the Linear R-D Model

By examining a large set of R-D parameters for many frames, we found that curves produced by the linear model has the following property: The slopes of the line segments form a monotonically increasing series, that is, $g_1 < g_2 < \cdots < g_z < 0$, where g_h is the slope of line segment in bitplane h Figure 4 demonstrates this property. This is intuitive because the most significant bitplanes carry higher significant bits, and thus the per-bit reduction in distortion is higher. This means that the most significant bitplanes will have steeper and more negative slopes.

While the above explanation of slope monotonicity makes sense, we have seen a few anomalies. For instance, frame 2 of Foreman has $g_1 = -0.15 > -1.8 = g_2$. Almost all anomalies occurred between bitplanes 1 and 2. From further examination, we found that the anomalies occur when the size of the first bitplane is very small, around 100 bytes or less . At that small size, the overhead of the bitplane header becomes significant, which negatively impacts the *effective* per-bit reduction in distortion (i.e., the slope) for bits in bitplane 1. We also notice that the smaller the size of bitplane 1, the higher the negative impact.

Because some of the proofs of our optimal bit allocation problem require the monotonicity of the slopes, we propose a simple processing of linear R-D curves to maintain monotonicity: we aggregate any two neighboring bitplanes into a single bitplane whenever there is a violation in monotonicity. For instance, the rightmost subfigure of Figure 4 shows the slopes before and after bitplane aggregation. Notice that if bitplane aggregation is needed, it is almost always sufficient to aggregate the first two planes, and in very rare cases between the second and third bitplanes. To verify this, we examined slopes of all bitplanes of all frames in four video sequences: Akiyo, Mother, Foreman, and Mobile. We found that only two pairs of consecutive bitplanes out of 5734 total bitplanes violated the monotonicity property. Furthermore, these two violations can be eliminated by two more bitplane aggregations (details are given in[5]).

Finally, we examined the impact of the bitplane aggregation on the accuracy of the *transformed* linear R-D curves. We repeat the accuracy assessment experiments in the previous subsection, but using linear R-D curves after bitplane aggregation is performed on them. Figure 3(c) shows that the average error between the modified linear R-D curves and the actual ones is the same as the (untransformed) linear R-D curves, which is less than 2% in almost all cases.

4. THE BIT ALLOCATION PROBLEM: FORMULATION AND SOLUTION

In this section, we consider the bit allocation problem for multi-sender video streaming systems that use FGS encoded streams. We assume the base layer is coded at a reasonably low bit rate and can be transmitted through a reliable channel. Our goal is to develop a practical algorithm for many-to-one streaming sessions such that the available resources are intelligently allocated to achieve the maximal perceptual streaming quality.

4.1. Problem Formulation

The bit allocation problem we address in this paper can be stated as follows. Given multiple senders that can potentially serve an FGS-encoded video sequence to a receiver, where senders have different portions of the sequence and have different outgoing bandwidth, and the receiver has a limit on the incoming bandwidth. Determine the streaming rate and the range of bits allocated to each sender to achieve the best possible video quality at the receiver. It is assumed that the base layer is coded at a low, and fixed bit rate and can be reliably delivered by any sender. We seek to optimize the streaming of the enhancement layer because it contributes significantly larger bit rates than the base layer, and unlike the base layer, its bit rate can be controlled.

We address this problem in two steps. First, we optimize the quality of individual frames of the sequence. That is, we formulate and solve the problem for each frame. In the second step, we divide the sequence into blocks of frames, each has a fixed number of frames. Then, we formulate and solve the optimization problem for each block. The second step is presented in the extended version of the paper.[5]

We now formulate the allocation problem for individual frames. Let T denote the frame period in seconds, which is the multiplicative inverse of the frame rate. T is fixed for all frames in the considered sequence. Assume there are n potential senders, each with outgoing bandwidth b_i ($i = 1, 2, \ldots, n$). Each sender i holds s_i contiguous bits, which is a portion of the enhancement layer bit stream. Without loss of generality, we assume that $s_1 \leq s_2 \leq \cdots \leq s_n$, otherwise we re-label senders to achieve that. Note that the nature of FGS-encoded sequences implies that a lower quality bit stream is always a subset of a higher quality one. That is, s_i is always a *prefix* of s_{i+1} for all $i = 1, 2, \ldots, n-1$. The receiver incoming bandwidth is denoted by b_I.

Solving the allocation problem should yield an allocation policy $A = \{(\Delta_i, r_i) | i = 1, 2, \ldots, n\}$, where Δ_i is the number of bits allocated to peer i and $r_i \leq b_i$ is its streaming rate. Bits are allocated to peers as follows. Peer 1 transmits the range from 0 to $\Delta_1 - 1$, peer 2 transmits from Δ_1 to $\Delta_1 + \Delta_2 - 1$, and in general peer i transmits from $\sum_{t=1}^{i-1} \Delta_t$ to $\sum_{t=1}^{i} \Delta_t - 1$. Mathematically, the bit allocation problem for a given frame can be formulated as:

$$\min_{A} \quad D(\sum_{t=1}^{n} \Delta_t) \tag{1a}$$

$$\text{s.t.} \quad \Delta_i - r_i T \leq 0 \tag{1b}$$

$$\sum_{t=1}^{i} \Delta_t \leq s_i \tag{1c}$$

$$r_i \leq b_i \tag{1d}$$

$$\sum_{t=1}^{n} r_t \leq b_I \tag{1e}$$

$$\Delta_i, r_i \in \mathbb{N}; \quad i = 1, 2, \ldots, n. \tag{1f}$$

The objective function in (1a) is to find the optimal allocation A^* that minimizes the reconstruction distortion, and hence maximizes the rendered quality. The constraints can be explained as follows: (1b) ensures that each sender has enough time to transmit all bits allocated to it ($\Delta_i - r_i T \leq 0 \implies \Delta_i / r_i \leq T$). (1c) ensures that the allocated bits to each sender are within the portion of bits stored at that sender, while (1d) and (1e) ensure that the limits on the incoming and outgoing bandwidth of the receiver and senders are not exceeded.

In order to solve this optimization problem, we need the mapping between the distortion and total number of bits received. Note that dividing the total number of bits by the (fixed) frame period T yields the bit rate. As we discussed in Section 3, we adopt the linear R-D model for this mapping because: (i) it is fairly accurate, (ii) its parameters can be efficiently extracted from the video sequence, and more importantly, (iii) it results in a linear objective function and hence efficient solution of the optimization problem. Recall that the linear R-D model divides the R-D curve into several line segments, each corresponds to a bitplane h and has a different slope g_h (see Figure 3(a)). The slope g_h represents the per-bit reduction in the distortion in bitplane h. Therefore, in order to compute the total distortion for a possible bit allocation, we need to find how many bits are transmitted from each bitplane. To do that, we introduce a new variable y_h ($h = 1, 2, \ldots, z$) to represent the number of bits transmitted from bitplane h. z is the number of bitplanes in the enhancement layer of the

considered frame. Now the distortion can be computed as $d + \sum_{h=1}^{z} g_h \; y_h$, where d is the distortion when only the base layer is transmitted. The optimization problem in (1) can be re-written as:

$$\min_{A} \quad D(\sum_{t=1}^{n} \Delta_t) = d + \sum_{v=1}^{z} g_v \; y_v \tag{2a}$$

$$\text{s.t.} \qquad \Delta_i - r_i T \leq 0 \tag{2b}$$

$$\sum_{t=1}^{i} \Delta_t \leq s_i \tag{2c}$$

$$r_i \leq b_i \tag{2d}$$

$$\sum_{t=1}^{n} r_t \leq b_I \tag{2e}$$

$$y_h \leq l_h \tag{2f}$$

$$\sum_{t=1}^{n} \Delta_t = \sum_{v=1}^{z} y_v \tag{2g}$$

$$\Delta_i, r_i, y_h \in \mathbb{N}; \;\; i = 1, 2, \ldots, n; \;\; h = 1, 2, \ldots, z. \tag{2h}$$

Notice the two new constrains in (2f) and (2g). (2f) makes sure that the number of bits transmitted from a bitplane does not exceed the size of that bitplane, whereas (2g) ensures that the total number of bits transmitted from different bitplanes is exactly the same as the number of bits allocated to senders.

Unlike (1), (2) is an integer linear programming (ILP) problem, because the objective function as well as all constraints are linear. While ILP problems are less complex than nonlinear problems, they are still NP-hard [16, page 777]. In the next section, we present a rounding scheme that transforms the ILP problem in (2) to a linear programming (LP) problem, which can be solved using the Simplex method or other efficient LP solvers. But before doing so, we need to make sure that the optimal solution for the ILP in (2) will produce a *valid* FGS-encoded bit stream. An FGS-encoded bit stream is valid if it has a contiguous stream of bits with no gaps between them. If there is a gap in the bit stream, the FGS decoder will ignore the rest of the bit stream beyond the gap, which will reduce the quality. The following lemma proves the validity of the optimal solution of (2).

LEMMA 1. *An optimal solution for (2) produces a contiguous FGS-encoded bit stream with no bit gaps.*

Proof. To prove this lemma, we need to show that no bits from a bitplane j will be transmitted before all bits of bitplanes $h < j$ are transmitted, where $h, j = 1, 2, \ldots, z$ and z is the number of bitplanes. That is, we need to show that if an optimal allocation results in $y_j^* \neq 0$ for any $j = 2, 3, \ldots, z$, then it must be the case that $y_h^* = l_h$, where $h = 1, 2, \ldots, j-1$, and l_h is the size of bitplane h. We prove this by contradiction.

Assume that the optimal allocation A^* produced $y_j^* \neq 0$ and there exists $y_h^* < l_h$ for some $h < j$. We construct another allocation $\widehat{A}$ which is exactly the same as A^*, except we shift $q = \min(y_j^*, l_h - y_h^*) > 0$ bits from bitplane j to bitplane h. $\widehat{A}$ is indeed a feasible allocation because it satisfies all the constraints in (2). Furthermore, the distortion associated with $\widehat{A}$ is given by $\widehat{D} = D^* + (g_h - g_j)q < D^*$, since g_h is always less than g_j because slopes are monotonically increasing, as discussed in Section 3. This is a contradiction because D^* is supposed to be the minimum distortion. □

4.2. Rounding Scheme and Linear Programming Solution

In this section, we present a rounding scheme that transforms the ILP problem in (2) to a linear programming (LP) problem, which can be solved using the Simplex method. We show that our rounding scheme results in a solution that is within a negligible gap from the optimal. Our LP formulation of the bit allocation problem is exactly the same as (2), except the constraint in (2h) is now relaxed to be:

$$\Delta_i, r_i, y_h \in \mathbb{R}^+ \cup \{0\}; \;\; i = 1, 2, \ldots, n; \;\; h = 1, 2, \ldots, z. \tag{3}$$

Let $\widehat{A} = \{(\widehat{\Delta}_i, \widehat{r}_i) | i = 1, 2, \ldots, n\}$ be the optimal allocation produced by solving the LP problem. $\widehat{\Delta}_i, \widehat{r}_i$ are in general non-negative real numbers. To obtain integer solutions for the original ILP problem, we propose the following rounding scheme:

$$\overline{r}_i = \lfloor \widehat{r}_i \rfloor, \quad \text{and} \quad \overline{\Delta}_i = \begin{cases} \lfloor \widehat{\Delta}_i \frac{\widehat{r}_i - 1}{\widehat{r}_i} \rfloor, & \widehat{r}_i > 1 \\ 0, & 0 \leq \widehat{r}_i \leq 1 \end{cases} \tag{4}$$

In the following two lemmas, we prove that this rounding scheme indeed produces a feasible solution, and that solution is very close to the optimal one.

LEMMA 2. *Rounding of the optimal solution of the relaxed linear programming problem using the rounding scheme in (4) always produces a feasible solution for the integer linear programming problem defined in (2).*

Proof. We only need to prove that constraint (2b) is satisfied, that is, we need to show that $\overline{\Delta}_i - \overline{r}_i T \leq 0$. Notice that all other constraints are automatically satisfied because we round down both of $\widehat{r}_i$ and $\widehat{\Delta}_i$.

We first consider the case when $\widehat{r}_i > 1$. For any $i = 1, 2, \ldots, n$, we have:

$$\frac{\overline{\Delta}_i}{\overline{r}_i} = \frac{\lfloor \widehat{\Delta}_i \frac{\widehat{r}_i - 1}{\widehat{r}_i} \rfloor}{\lfloor \widehat{r}_i \rfloor} \leq \frac{\widehat{\Delta}_i \frac{\widehat{r}_i - 1}{\widehat{r}_i}}{\lfloor \widehat{r}_i \rfloor} < \frac{\widehat{\Delta}_i \frac{\widehat{r}_i - 1}{\widehat{r}_i}}{\widehat{r}_i - 1} = \frac{\widehat{\Delta}_i}{\widehat{r}_i} \leq T,$$

where the first inequality (from the left) is from the definition of the floor function, and the second inequality is due to the fact that $\lfloor \widehat{r}_i \rfloor > \widehat{r}_i - 1$. Next we consider the case when $0 \leq \widehat{r}_i \leq 1$. In this case, $\overline{\Delta}_i = 0$. Therefore, $\overline{\Delta}_i - \overline{r}_i T = -\overline{r}_i T \leq 0$, because $\overline{r}_i \geq 0$, and the constraint is satisfied. □

LEMMA 3. *The rounding scheme in (4) results in a total number of bits that is smaller than the optimal number of bits by at most $nT + n$, where n is the number of senders and T is the frame period.*

Proof. We first compute the gap between the optimal and rounded solutions for an arbitrary sender i, where $i = 1, 2, \ldots, n$. If $\widehat{r}_i > 1$, we have:

$$\widehat{\Delta}_i - \overline{\Delta}_i = \widehat{\Delta}_i - \lfloor \widehat{\Delta}_i \frac{\widehat{r}_i - 1}{\widehat{r}_i} \rfloor < \widehat{\Delta}_i - \widehat{\Delta}_i \frac{\widehat{r}_i - 1}{\widehat{r}_i} + 1 = \frac{\widehat{\Delta}_i}{\widehat{r}_i} + 1 \leq T + 1.$$

If $0 \leq \widehat{r}_i \leq 1$, we have

$$\widehat{\Delta}_i - \overline{\Delta}_i = \widehat{\Delta}_i \leq \widehat{r}_i T \leq T < T + 1.$$

Therefore, the gap for any sender is bounded by $T + 1$. Thus for all n senders, the gap is at most $nT + n$. □

Lemma 3 shows that the gap between the solution resulted from our rounding scheme and the optimal one is negligible. To put this gap into perspective, consider an extreme-case streaming session in which 30 senders are concurrently streaming to a single receiver. Assume that the frame rate is 15 frames per second. Then, the gap is at most $30/15 + 30 = 32$ bits. That is, the rounded solution may transmit at most 4 bytes less than the optimal one from any given frame, which is indeed negligible given that frame sizes are in the order of kilo bytes.

5. FGSASSIGN: AN OPTIMAL ALGORITHM FOR THE SINGLE-FRAME BIT ALLOCATION PROBLEM

The LP formulation of the bit allocation problems, presented in the previous section, can be solved by the Simplex method. On average, the Simplex method is efficient, but it has a worst-case exponential running time [17, Section 8.6]. While there exist worst-case polynomial time methods (e.g., Karmarkar's interior-point algorithm)[18] for solving general LP optimization problems, they are quite complex to implement, and they have very large average running times, which in many cases exceed the average running time of the Simplex method.

We propose a greedy algorithm, called FGSAssign, to efficiently solve the single frame bit allocation problem. The pseudo code for the algorithm is given Figure 5. The basic idea of the algorithm is to transmit the maximum possible number of bits from each senders. It does so by first sorting all senders based on the portion of the stream stored at each sender, such that $s_1 \leq s_2 \leq \cdots \leq s_n$. Then, it sequentially allocates to sender i $(i = 1, 2, \ldots, n)$ the maximum number

FGSAssign

1. Sort all senders based on s_i, where $s_1 \leq s_2 \leq \cdots \leq s_n$;
2. $x_0 = \cdots = x_n = 0$; $\Delta_1 = \cdots = \Delta_n = 0$; $r_{agg} = 0$;
3. **for** $i = 1$ to n **do**
4. $x_i = \min(x_{i-1} + b_i T, s_i)$;
5. $r_i = (x_i - x_{i-1})/T$;
6. **if** $(r_{agg} + r_i < b_I)$ **then**
7. $r_{agg} = r_{agg} + r_i$;
8. $\Delta_i = x_i - x_{i-1}$;
9. **else**
10. $r_i = b_I - r_{agg}$;
11. $\Delta_i = T \times r_i$;
12. **return**
13. **endfor**

Figure 5. Pseudo code for an optimal and efficient algorithm for the single-frame bit allocation problem.

of bits which sender i can transmit within the frame period T. The allocated bits must be available at sender i, i.e., they are a subset of s_i, and they do not overlap with bits allocated to j $(1 \leq j < i)$. Thus FGSAssign avoids holes as well as overlapping of bits among senders. Therefore, the allocation will result in a contiguous bit stream, which is necessary for decoding FGS-encoded enhancement layers. The following theorem proves that FGSAssign is optimal and efficient.

THEOREM 1. *The FGSAssign algorithm terminates in $O(n \log n)$ steps, where n is the number of senders, and it produces an allocation that minimizes the distortion of individual frames.*

Proof. The termination and time complexity part is straightforward: sorting of s_i's takes $O(n \log n)$ steps and the for-loop iterates up to n times, each taking a constant number of operations.

For the optimality part, we first notice that for individual frames, sending more bits always results in smaller distortion. Therefore, sending the maximum number of bits corresponds to the minimum distortion. The maximum number of bits is determined based on either: (i) the receiver incoming bandwidth b_I, or (ii) senders outgoing bandwidth b_i and portions of stream stored s_i $(1 \leq i \leq n)$. In the first case, because s_i's are sorted, the algorithm fills up the entire receiver's bandwidth with non overlapping bits and returns from the for-loop in line 12. Thus the maximum number of bits will be transmitted.

In the second case, the algorithm terminates after finishing n iterations of the for-loop. In every iteration i of the for-loop, the algorithm assigns to sender i the maximum number of bits that this sender can transmit, constrained only by the outgoing bandwidth b_i and the length of the stored portion of the stream s_i. In other words, the algorithm makes a greedy decision (Δ_i, r_i), and solves a subproblem in the $i+1$ iteration. To prove that this greedy approach is optimal, we show that the problem has two properties: greedy-choice and optimal substructure [16, Chapter 16].

The greedy-choice property guarantees that a globally optimal solution can be arrived at by a greedy choice. Suppose $\{(\Delta_u^*, r_u^*) | u = i, i+1, \ldots, n\}$ is an optimal solution to the original problem in iteration i. Since (Δ_i, r_i) is a greedy choice, we have $\Delta_i \geq \Delta_i^*$ (and $r_i \geq r_i^*$). Now, we construct a new solution from the optimal one by shifting bits from other peers to i, such that the new solution contains the greedy decision. Obviously, the number of bits remains the same, thus, we have an optimal solution consists of the greedy choice.

The optimal substructure property ensures that combining a greedy decision with an optimal solution to the subproblem results in an optimal solution to the original problem. We prove this property by contradiction. Consider iteration i. Suppose the combination of the greedy decision (Δ_i, r_i) and a subproblem optimal solution $\{(\Delta_v, r_v) | v = i+1, i+2, \ldots, n\}$ is not an optimal solution to the original problem. Thus, we can find an optimal solution $\{(\Delta_u^*, r_u^*) | u = i, i+1, \ldots, n\}$ such that $\sum_{u=i}^{n} \Delta_u^* > \sum_{v=i}^{n} \Delta_v$. Since $\Delta_i \geq \Delta_i^*$ by the greedy decision, we must have $\sum_{u=i+1}^{n} \Delta_i^* > \sum_{v=i+1}^{n} \Delta_v$, which is a contradiction because $\sum_{v=i+1}^{n} \Delta_v$ is assumed to be an optimal solution to the subproblem. □

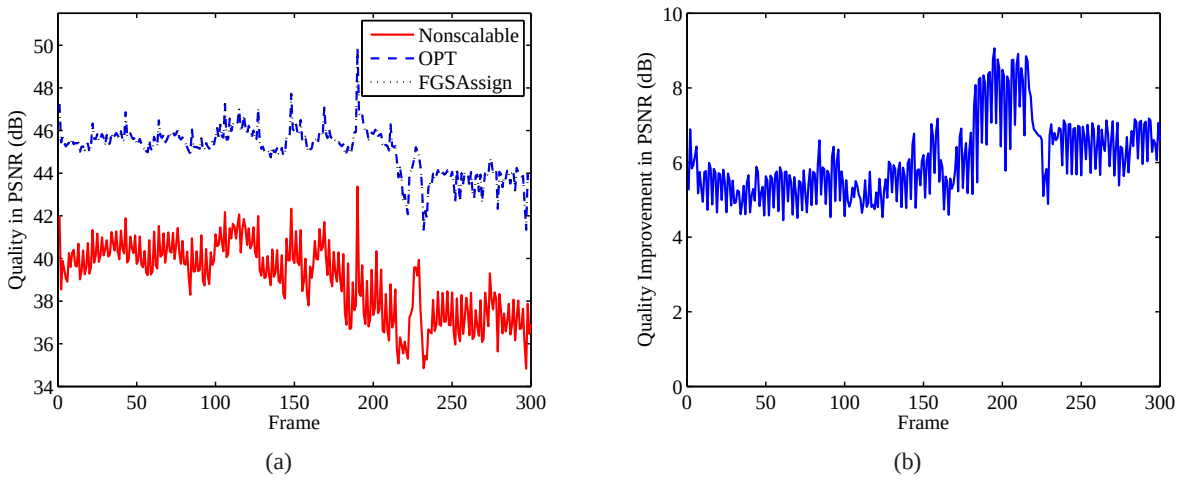


Figure 6. Comparison between nonscalable, OPT, and FGSAssign algorithms: (a) Quality achieved by each algorithm, and (b) Potential quality improvement (in dB) by using scalable over nonscalable algorithms. Data shown for streaming Foreman sequence in scenario I.

6. EVALUATION

In this section, we evaluate the importance of the scalable allocation problem, and we verify the optimality of our FGSAssign algorithm.

6.1. Experimental Setup

Software used and developed. In our experiments, we use the MPEG-4 Reference Software Version 2.5[19] developed by Microsoft as an experimental package for the MPEG-4 standard. It is implemented in C++ and contains three major executables: *encoder*, *decoder*, and *fgs_server*. We instrument the reference software to extract various statistics of a video sequence. For instance, we collect the transform coefficients, number of bitplanes, and size of each bitplane in the enhancement layer. This information is used to estimate the parameters of the linear R-D model.

We have implemented the FGSAssign algorithm (Figure 5) and the optimal algorithm (referred to as OPT) using the Simplex method. For comparisons, we have also implemented the nonscalable allocation algorithm discussed in Section 1. All allocation algorithms are implemented in Matlab. We run the experiments on a 3.0 GHz Pentium 4 workstation running Windows XP.

Streaming scenarios and video test sequences. To compare the performance of the allocation algorithms, we design four representative streaming scenarios, which we believe capture various Internet and Intranet streaming settings. In the first scenario, we consider a receiver using a high-speed connection with enough incoming bandwidth to receive full quality stream, and four senders with 512 kbps, 256 kbps, 1.5 Mbps, and 3 Mbps outgoing bandwidth. The two senders with lower outgoing bandwidth represent peers with cable modem or ADSL connections, while the other two could be office or campus workstations. Due to the asymmetry between incoming and outgoing bandwidth, peers typically receive video streams with higher bit rates (quality) than they could serve to others. Therefore, we assume that the senders store different version of the FGS-encoded stream: 512 kbps, 4 Mbps, 8 Mbps, and 10 Mbps, respectively.

The second scenario assumes the receiver has 1 Mbps incoming bandwidth, and there are six senders. Four of them subscribe to 128 kbps uplink access service while the other two only have 64 kbps uplink. Among those four 128 kbps senders, three of them store 1 Mbps version of the coded stream, and the last one stores a 512 kbps version. All others have a 128 kbps version of the stream. The third scenario simulates an Intranet video casting that is very common nowadays. Suppose there are four senders at different facilities, each of them has 1.5 Mbps connection. The receiver, located at a regional center, has larger bandwidth of 3 Mbps. Senders store four different versions of the stream: 256 kbps, 512 kbps, 4 Mbps, and 4Mbps, respectively. The forth scenario has five senders: four of them have 256 kbps outgoing bandwidth and store a 1.5 Mbps coded stream. The last sender has 512 kbps outgoing bandwidth and holds a 512 kbps coded version of the stream. The receiver has 1.5 Mbps incoming bandwidth.

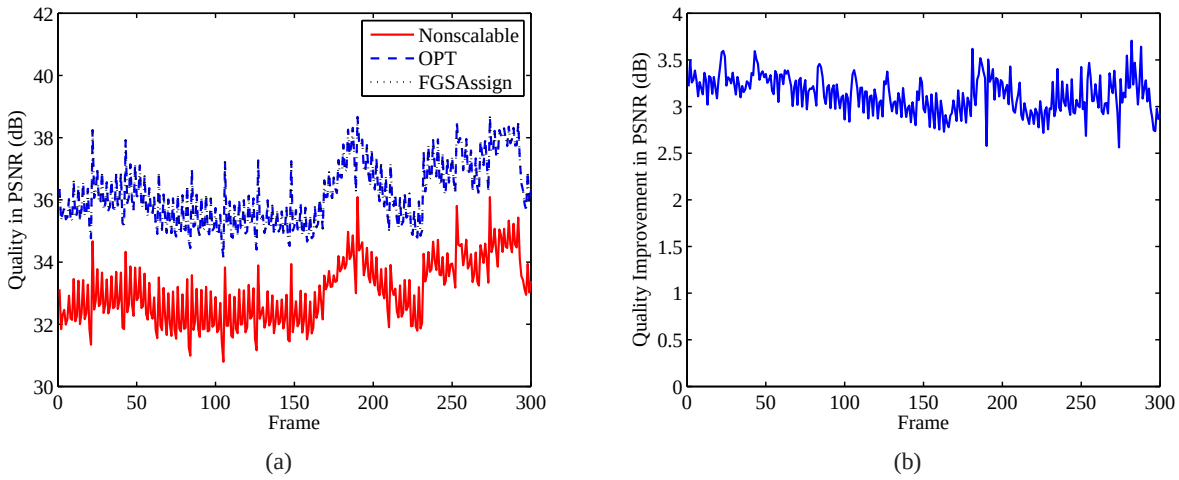


Figure 7. Comparison between nonscalable, OPT, and FGSAssign algorithms: (a) Quality achieved by each algorithm, and (b) Potential quality improvement (in dB) by using scalable over nonscalable algorithms. Data shown for streaming Mobile sequence in scenario III.

In all scenarios, we stream a set of video sequences with different characteristics. To form this set of test sequences, we have analyzed twenty representative video sequences from various sources. We categorize sequences into three complexity classes based on the average spatial and temporal complexities of each sequence.[13] We present results for two sequences: Foreman and Mobile. Both are coded at 30 fps, and have a CIF (352x288) resolution. Foreman sequence is a low-complexity sequence and has a scene cut, where the second scene has more details than the first one. Mobile contains saturated colors and several moving objects, and therefore is a high-complexity sequence.

6.2. Results

We present a sample of our results in this section due to space limitations, more results are available in.[5]

Scalable versus nonscalable allocation. Our results clearly confirm the potential quality improvement from using scalable allocation algorithms over the nonscalable ones: In all considered streaming scenarios and with all test sequences, there was at least 1 dB and up to 8 dB improvement in quality (see plots for all cases in[5]). Figure 6 shows a sample plot for streaming of Foreman in scenario I, where we see an average quality improvement of at least 5 dB. Figure 7 demonstrates streaming of Mobile in scenario III. The figure shows more than 3 dB quality improvement.

Optimality of FGSAssign. To verify our analytic results regarding the optimality of the FGSAssign, we compare the distortion achieved by solving the LP problem using the Simplex method versus the distortion achieved by the FGSAssign algorithm. In all cases, FGSAssign produced exactly the same distortion values as the optimal as indicated in Figures 6(a) and 7(a).

7. CONCLUSION

In this paper, we formulated and solved the bit allocation problem for FGS-encoded video sequences streamed in distributed and heterogeneous environments. The formulation was done in a number of steps. First a general optimization problem was formed, which is transformed to an integer linear programming (ILP) one. Then using a simple rounding scheme, the ILP problem is transformed to a linear programming problem. We proposed an optimal allocation algorithm (FGSAssign) for the single-frame allocation problem. Our experimental results show that solving the bit allocation problem using our algorithm could provide significant quality improvement over solving it using the nonscalable algorithm: An improvement of up to 8 dB could be achieved in some cases. The quality improvement is even higher when allocating bits to blocks of multiple frames. This is because multiple frame allocation provides more chances for optimization.

Acknowledgments

This research is partially supported by an NSERC Discovery Grant and by a President's Research Grant from Simon Fraser University.

REFERENCES

1. M. Hefeeda, A. Habib, D. Xu, B. Bhargava, and B. Botev, "CollectCast: A peer-to-peer service for media streaming," *ACM/Springer Multimedia Systems Journal* **11**, pp. 68–81, November 2005.
2. N. Magharei and R. Rejaie, "Adaptive receiver-driven streaming from multiple senders," *ACM/Springer Multimedia Systems Journal* **11**, pp. 1–18, April 2006.
3. Y. Cui and K. Nahrstedt, "Layered peer-to-peer streaming," in *Proc. of ACM International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'03)*, (Monterey, CA), June 2003.
4. T. Nguyen and A. Zakhor, "Distributed video streaming over Internet," in *Proc. of ACM/SPIE Multimedia Computing and Networking (MMCN'02)*, (San Jose, CA), January 2002.
5. C. Hsu and M. Hefeeda, "Optimal bit allocation for fine-grained scalable video sequences in distributed streaming environments," Tech. Rep. TR 2006-20, Simon Fraser University, July 2006. Available online at http://nsl.cs.surrey.sfu.ca/projects/fgs/.
6. X. Su and T. Wang, "Sequence of linear programming for transmission of fine-scalable coded content in bandwidth-limited environments," *ACM/Springer Multimedia Systems Journal* **11**, pp. 455–466, June 2006.
7. P. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," *IEEE Transactions on Multimedia* **8**, pp. 390–404, April 2006.
8. J. Chakareski and B. Girod, "Rate-distortion optimized packet scheduling and routing for media streaming with path diversity," in *Proc. of Data Compression Conference (DCC'03)*, (Snowbird, UT), March 2003.
9. A. Begen, Y. Altunbasak, and M. Begen, "Rate-distortion optimized on-demand media streaming with server diversity," in *Proc. of IEEE International Conference on Image Processing (ICIP'03)*, (Barcelona, Spain), September 2003.
10. M. Dai and D. Loguinov, "Analysis of rate-distortion functions and congestion control in scalable Internet video streaming," in *Proc. of ACM International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'03)*, (Monterey CA), June 2003.
11. M. Dai, D. Loguinov, and H. Radha, "Rate-distortion modeling of scalable video coders," in *Proc. of IEEE International Conference on Image Processing (ICIP'04)*, (Singapore), October 2004.
12. J. Sun, W. Gao, D. Zhao, and Q. Huang, "Statistical model, analysis and approximation of rate-distortion function in MPEG-4 FGS videos," in *Proc. of SPIE International Conference on Visual Communication and Image Processing (VCIP'05)*, (Beijing, China), July 2005.
13. C. Hsu and M. Hefeeda, "On the accuracy and complexity of rate-distortion models for fine-grained scalable video sequences," Tech. Rep. TR 2006-12, Simon Fraser University, August 2006. Available online at http://nsl.cs.surrey.sfu.ca/projects/fgs/.
14. Y. Wang, J. Ostermann, and Y. Zhang, *Video Processing and Communications*, Prentice Hall, 2002.
15. X. Zhang, A. Vetro, Y. Shi, and H. Sun, "Constant quality constrained rate allocation for FGS-coded video," *IEEE Transactions on Circuits and Systems for Video Technology* **13**, pp. 121–130, February 2003.
16. T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*, MIT Press, 2nd ed., 2001.
17. C. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*, Dover, 1st ed., 1998.
18. D. Goldfarb and M. Todd, "Linear programming," *Handbook in Operations Research and management Science, Optimization* **1**, pp. 73–170, 1989.
19. ISO/IEC 14496-5, "MPEG-4 Visual reference software," February 2004.