Rate-Distortion Based Scheduling of Video with Multiple Decoding Paths

Huisheng Wang and Antonio Ortega *

Integrated Media Systems Center,
Department of Electrical Engineering-Systems,
University of Southern California, Los Angeles, CA 90089
{huishenw,ortega}@sipi.usc.edu

ABSTRACT

We present a general rate-distortion based scheduling framework that can accommodate cases where multiple encoded versions for the same video are available for transmission. Previous work on video scheduling is mostly focused on those encoding techniques, such as layered coding, which generate only one set of dependent packets. However, it is sometimes preferred to have a codec that produces redundant video data, where multiple different decoding paths are possible. Examples of these scenarios are multiple description layered coding and multiple independently encoded video streams. A new source model called Directed Acyclic HyperGraph (DAHG) is introduced to describe the relationship between different video data units with multiple decoding paths. Based on this model, we propose two low-complexity scheduling algorithms: the greedy algorithm and the M-T algorithm. Experiments are made to compare the performance of these algorithms. It is shown that, in the case of multiple decoding paths, the M-T algorithm outperforms the greedy algorithm by taking into account some of the transmission possibilities available in the near future before making a decision.

Keywords: multimedia communication, rate-distortion based scheduling, multiple decoding paths, multiple description layered coding

1. INTRODUCTION

Recent technological developments and the rapid growth of Internet and wireless networks make it feasible and more attractive to provide network-based real-time video services. These networks are characterized by variations in bandwidth, delay and packet loss rate, which can severely affect the reproduction quality of the video delivered through the network. The basic goal of video scheduling is to maximize the playback quality at the decoder, by adapting to the changing network conditions and application requirements. Previous work on video scheduling [1–3] is mostly focused on those encoding techniques, such as layered coding, that generate only one set of dependent packets. However, it is sometimes impractical for the encoder to adapt to the varying channel conditions on the fly, as on short notice an encoder may have to completely switch between different modes of operation. Thus, in our proposed framework, we assume that the codec produces redundant video data, where multiple different decoding paths are possible. The decisions at run-time determine which of those decoding paths will be followed and what should be transmitted. Examples of these scenarios are multiple description layered coding [4] and multiple independently encoded video streams [5].

The scheduling problem has been studied for the case of layered coding [1–3]. Chou and Miao [1] proposed a rate-distortion optimized framework for packet scheduling over a lossy packet network based on Lagrangian optimization. Miao and Ortega [2, 6] simplified [1] by proposing a greedy solution that explicitly considers the effects of data dependencies and delay constraints and combines them into a single importance metric. All these algorithms are based on a simple source model, a Directed Acyclic Graph (DAG), that only considers one decoding choice (i.e., a single decoding path): a packet can be decoded only when all of its dependent data units are received and decodable. Implicitly this approach excludes the possibility of having multiple descriptions, in which several decoding choices are possible based on which descriptions are received at the receiver. Based on a

*This research has been funded in part by the Integrated Media Systems Center, a National Science Foundation Engineering Research Center, Cooperative Agreement No. EEC-9529152.

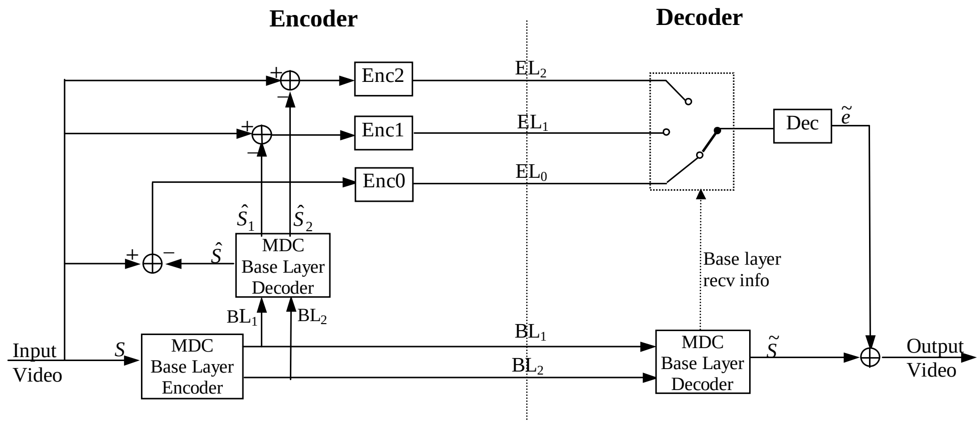

Figure 1. Structure of the MDLC codec.

DAG as well, Cheung and Tan introduced a more general formulation that a data unit may be decoded even when only part of its dependent data units are available and the distortion reduction of the data unit is a function of its available dependent data units. In this work all possibilities of decoding and delivery scenarios are considered, which leads to substantial increases in complexity. In our approach we propose a more structured source model that can consider multiple decoding paths and also allow in practical cases a "pruning" or "merging" of certain decoding paths, thus enabling a reduced complexity search.

In this paper, we extend our work for multiple description layered coding (MDLC) in [4] to a more general class of problems where multiple decoding paths exist. This includes applications where error resilience tools are available or where multiple redundant encoded versions of the input signal can be used for transmission. We first introduce a new Directed Acyclic HyperGraph (DAHG) to represent the data dependencies and correlation between different video data units. The expected end-to-end distortion for a group of packets can then be estimated based on this model. Though we can directly apply the Lagrangian optimized approach in [1] with the expected distortion expressions we derive, it may be too complex, especially if this system needs to operate at a resource-constrained server for multiple decoding scenarios at run-time. Following the analysis of the Taylor expansion of the expected distortion, we propose two low-complexity scheduling techniques: the greedy algorithm and the M-T algorithm. The proposed scheduling algorithms based on a DAHG take into account the impact of delay constrained delivery, channel conditions, and data dependencies and correlation between data units. Here, we use the MDLC in [4] as an example to describe our scheduling algorithms, as well as for our experiments.

This paper is organized as follows. We briefly review the MDLC codec [4] in Section 2. Section 3 describes the DAHG source model, its associated parameters and the expected end-to-end distortion. In Section 4, we derive Taylor expansion of the expected distortion and use this to compare the behavior of codecs with only a single decoding path to that of allowing multiple decoding paths. This leads to our low-complexity scheduling algorithms in Section 5. The simulation results are presented in Section 6. Finally we conclude our work in Section 7.

2. MULTIPLE DESCRIPTION LAYERED CODING

The MDLC system we proposed in [4] combines the hierarchical scalability of layered coding (LC) with the reliability of multiple description coding (MDC). Fig. 1 shows the structure of the MDLC codec. The MDLC coder uses an MDC encoder to generate two base layer descriptions BL_1 and BL_2. Then the base layer MDC decoder in the MDLC encoder module replicates the three possible decoding scenarios at the receiver: both descriptions received or either one received. If both descriptions are received, the base layer $\hat{S}$ is generated, and the difference between the original video input S and $\hat{S}$ is coded with a standard encoder (e.g., MPEG-4 FGS) into an enhancement layer stream EL_0. If only one description is received, the base layer decoder generates a low quality reproduction, $\hat{S}_1$ or $\hat{S}_2$, and feeds the differences $S - \hat{S}_i$, $i = 1, 2$, into two enhancement layer encoders separately to create EL_1 and EL_2. The decoder system is composed of two parts: base layer MDC

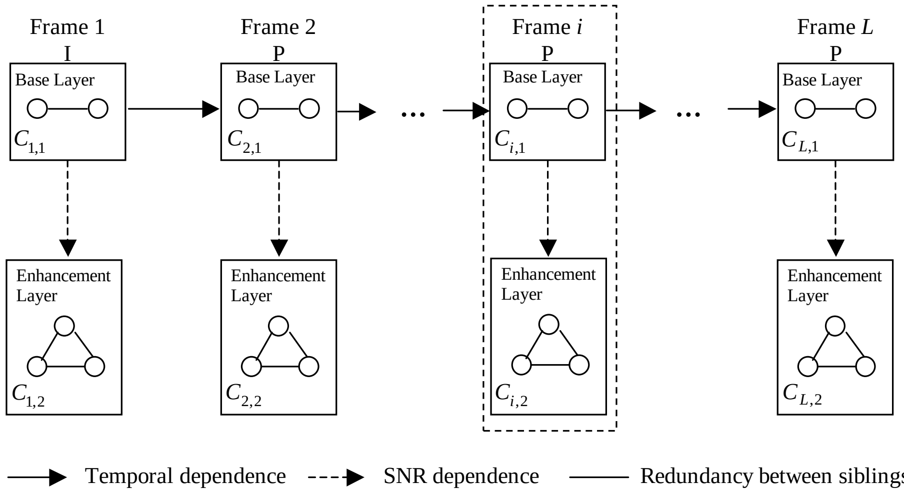

Figure 2. An example DAHG of the MDLC system for a group of I-P frames.

decoder, and enhancement layer switch and decoder. The base layer MDC decoder will generate a reproduction $\tilde{S}$ which is $\hat{S}$, $\hat{S}_1$ or $\hat{S}_2$ depending on what was received. The enhancement layer switch then selects which EL stream to decode given what base layer was received. Thus multiple decoding paths exist for the enhancement layer, including the case that both BL_1 and BL_2 are received, and the two cases that either one of them is received. Finally, the decoded base layer and enhancement layer will be combined together to generate the final video output. Here there are combinations of transmitted data such that future transmissions may reduce the importance value of current ones. For example, if we sent both BL_1 and BL_2, and then if BL_1 is received, BL_2 is worthless.

3. SOURCE MODEL

3.1. Directed Acyclic Hypergraph (DAHG)

Previous research [1, 2] modelled the source dependencies between a group of data units as a DAG. Though this graph model leads to a simple and accurate representation for a LC system, it cannot represent more general cases such as multiple descriptions in a MDC or MDLC system, where multiple different decoding paths are possible. Here we introduce a new model called Directed Acyclic HyperGraph (DAHG) to present the relationship between different video data units when multiple encoded versions are available for the same video. A DAHG is similar to a normal DAG, except that each vertex is composed of a clique which contains a set of nodes and every pair of nodes is connected by an undirected edge. A clique corresponds to a logical source coding unit that is encoded into multiple versions. Each version is represented by a node or data unit. There are two kinds of edges in a DAHG: directed edges indicate a dependence relation, and undirected edges indicate a redundancy relation. A clique C is called a *parent* of clique C' if there is a directed edge pointing from C to C'. Then C' is called a *child* of C. C' can be decoded only if all of the input edges from its parents including C are activated, which means at least one of the nodes of each of its parents must be decodable in order to decode C'. The nodes inside a clique have redundancy between them, and they are called *siblings* of each other with undirected edges connecting them. We write $i \prec j$ if object i is a parent of j, $i \succ j$ if i is a child of j, and $i \sim j$ if i and j are siblings.

Fig. 2 shows an example DAHG for the MDLC system we reviewed in Section 2. Each frame i contains a base layer clique C_{i1} and an enhancement layer clique C_{i2}. A directed edge is used to represent both the SNR dependence between C_{i1} and C_{i2}, and the temporal dependence between two cliques of different frames if they are temporally dependent. C_{i1} has two nodes representing the two base layer descriptions BL_1 and BL_2 respectively. Similarly, there are three nodes in C_{i2} corresponding to EL_0, EL_1 and EL_2. The distortion reduction of C_{i2} then depends on which subset of nodes in C_{i1} are received.

3.2. Static Parameters Associated with DAHG

Each object in DAHG has several static parameters associated with it . Some of these parameters are constants measured at the encoding stage. As in [1], each data unit l has a size r_l in bytes and a time deadline t_l by which it must arrive at the receiver to be useful for decoding.

However, the distortion reduction of a data unit in the DAHG can take different values depending on the decoding path in which it will be decoded. We first introduce the concepts of a clique state and a decoding path. Assume that a clique contains N data units. Since each unit can be either received correctly or not received (due to loss or because it was not transmitted in the first place), there is a total of 2^N possible states for the clique. A clique state is represented by a length-N binary string s, with each bit indicating the receiving status of a data unit in the clique. Let b_l denote the corresponding bit location of data unit l in s. Therefore, the b_lth bit of s is 1 (written as $s[b_l] = 1$ in mathematics) if l arrives at the receiver on time and is 0 otherwise. A decoding path leading to clique C is defined as a particular combination of all C's parent clique states including both direct or indirect parents. Strictly speaking, this definition of a decoding path does not work for those cliques that do not have parents. Once a node in this class of cliques is received at the decoder, it can be decoded independently regardless of the states of other cliques. Therefore, in order to use the same mathematical notation, we simply assume there is one virtual decoding path leading to those cliques, and this path always exists. Let Q_C be the set of all decoding paths leading to C. Then we can represent the distortion reduction of data unit l in the clique by a distortion vector $\mathbf{d}_l = [d_l^{(1)}, d_l^{(2)}, ..., d_l^{(q)}, ..., d_l^{(|Q_c|)}]$, where $d_l^{(q)}$ is the distortion reduction if l is decoded in the qth decoding path, and $|.|$ denotes the cardinality of the set.

Though each of the data units in clique C can produce a certain amount of distortion reduction, the total distortion reduction when more than one data units are received correctly is usually less than the sum of their respective distortion reductions. We introduce the redundancy matrix $\mathbf{I}_C$, to represent the redundancy between different data units inside the same clique C. $\mathbf{I}_C = [I_C^{(s,q)}]_{|S_C| \times |Q_C|}$, where S_C is the set of all clique states in C, and $I_C^{(s,q)}$ is the redundancy of C when it is in state s, and decoded in the qth decoding path. $I_C^{(s,q)}$ is given by

$$I_C^{(s,q)} = \sum_{l \in B_C^{(s)}} d_l^{(q)} - d_C^{(s,q)} \tag{1}$$

Where $B_C^{(s)} = \{l | l \in C \ \& \ s[b_l] = 1\}$ represents the set of data units in C that are received in state s, and $d_C^{(s,q)}$ is the total distortion reduction of C if it is decoded in state s and in the qth decoding path.

Fig. 3 lists the distortion vectors and redundancy matrices for an I frame of an MDLC example, whose DAHG model is shown in Fig. 2. The base-layer clique C_1 of an I-frame has no parents, and thus the distortion vector of each base-layer node has only one element d_1^b by the assumption of having one virtual decoding path. The redundancy between BL_1 and BL_2 exists when both of them are received, i.e., at state s_{C1} =[1 1]. An enhancement-layer clique C_2 can produce distortion reduction in three different decoding paths corresponding to s_{C1} =[1 0], [0 1] and [1 1], respectively. The distortion reduction of each data unit in clique C_2 depends on the decoding path. For example, EL_1 can reduce the distortion by d_1^e when BL_1 is received, i.e. the base-layer clique C_1 is at state [1 0] or [1 1]. However, if only BL_2 is received at state s_{C1} =[0 1], EL_1 cannot be decoded and therefore the distortion reduction of EL_1 in this decoding path is 0. The redundancy matrix of clique C_2 is a 4×3 matrix, with each component representing the redundancy of C_2 at a certain state s_{C2} and a given decoding path based on state s_{C1}. The DAHG model associated with the distortion-redundancy representation can also be applied in the general setting, such as error resilience, for which we can add a second node in each clique with size zero but having certain distortion reduction if previous frames are received.

In a packet-switching network, all the data are packetized before transmission. A data unit can be either put into one packet or split into several packets. When splitting happens, the distortion related parameters (distortion vector and redundancy matrix) are assigned to each packet differently according to their coding methods. In the case that a data unit can only be decoded when all its packets are received, for example the base layer node, the scheduling algorithm will treat, as a whole, as if it was not split at all. On the other hand, a data unit may be coded as an embedded stream that can be truncated into any number of bits within this stream, to provide partial distortion reduction proportional to the number of bits decoded for this unit. The

MDLC example: A base-layer clique C_1 contains data units BL_1 and BL_2, and an enhancement-layer clique C_2 contains EL_1, EL_2, and EL_0. The following assumptions are made on the coding algorithm:

- The two descriptions are symmetric, i.e., the distortion values for (BL_1, EL_1) and (BL_2, EL_2) are the same.
- The cross-description decoding of EL_2 given BL_1 or EL_1 given BL_2 is ignored since the information added by the cross enhancement layer is very small.
- The total distortion reduction provided by the following three cases is the same: (1) BL_1 and EL_1, (2) BL_2 and EL_2, and (3) BL_1, BL_2 and EL_0.

Then we can assign the following parameters to each clique and data unit.

(1) Base layer: Clique state $s_{C1} = [BL_1, BL_2]$

Distortion vector: $\mathbf{d}_{BL1} = \mathbf{d}_{BL2} = [d_1^b]$,

Redundancy matrix: $I_{C1}(s_{C1} = [1\ 1]) = [2d_1^b - d_0^b]$,

Where d_1^b is the base-layer side distortion, and d_0^b is the base-layer center distortion when both BL_1 and BL_2 are received.

(2) Enhancement layer: Clique state $s_{C2} = [EL_1, EL_2, EL_0]$

Distortion vector: each column describes a different decoding path depending on the state of C_1

	$s_{C1} = [1\ 0]$	$s_{C1} = [0\ 1]$	$s_{C1} = [1\ 1]$
$\mathbf{d}_{EL1}$	d_1^e	0	d_0^e
$\mathbf{d}_{EL2}$	0	d_1^e	d_0^e
$\mathbf{d}_{EL0}$	$d_0^e/2$	$d_0^e/2$	d_0^e

Redundancy matrix:

I_{C2}	$s_{C1} = [1\ 0]$	$s_{C1} = [0\ 1]$	$s_{C1} = [1\ 1]$
$s_{C2} = [1\ 1\ 0]$	0	0	d_0^e
$s_{C2} = [1\ 0\ 1]$	$d_0^e/2$	0	d_0^e
$s_{C2} = [0\ 1\ 1]$	0	$d_0^e/2$	d_0^e
$s_{C2} = [1\ 1\ 1]$	$d_0^e/2$	$d_0^e/2$	$2\,d_0^e$

Figure 3. The distortion related quantities assigned for the I-frame of an MDLC example, whose DAHG model is shown in Fig. 2.

enhancement layer in the FGS coding is such an example. In this case, a node contains several packets, and each packet can only be decoded if all its dependent packets in the node are decodable. We then assign each packet p a distortion ratio $\rho_p = \frac{1}{|Q_C|}\sum_{q\in Q_C}\frac{d_p^{(q)}}{d_l^{(q)}}$, where $d_p^{(q)}$ is the partial distortion reduction contributed by packet p in the qth decoding path, and $d_l^{(q)}$ is the total distortion reduction of data unit l. Though the actual ratio between $d_p^{(q)}$ and $d_l^{(q)}$ may vary slightly with different decoding paths, we use the average ratio as an approximation for simplicity.

3.3. Expected End-to-End Distortion

Suppose we wish to transmit a group of L packets whose time deadlines fall in a time window. The relations between these packets are specified by a DAHG model. We now try to estimate the expected end-to-end distortion of this group of packets (GOPkt) when given a vector of packet loss probability (PLP) providing a loss probability for each packet in the group. Note that a packet is considered lost if it is either lost or arrives at the decoder too late to be played. At a given transmission time, we define the "transmission state" as the PLP vector which accounts for the transmission history of the GOPkt up to the current time under the past channel conditions. Let ϵ_l be the PLP of packet $l \in \{1, ..., L\}$ and let $\boldsymbol{\epsilon} = [\epsilon_1, ..., \epsilon_L]$ be the real-time transmission state. The expression of the expected distortion in a DAHG differs from that in [1] mainly in two ways: (1) multiple decoding paths in a DAHG vs. single decoding path in a DAG, and (2) multiple decodable clique states in a DAHG vs. 0/1 events of the data unit (i.e., it is either decodable or not) in a DAG. The incrementally additive distortion model of the DAG holds for cliques in the DAHG, but not for nodes any more.

To help us write an expression of the expected distortion, we first derive some related probabilities. The probability of occurrence of clique state s is given by

$$p_C^{(s)} = \prod_{l \in B_C^{(s)}} (1-\epsilon_l) \prod_{l \in \bar{B}_C^{(s)}} \epsilon_l \tag{2}$$

where $B_C^{(s)} = \{l | l \in C \ \& \ s[b_l] = 1\}$, and $\bar{B}_C^{(s)} = \{l | l \in C \ \& \ s[b_l] = 0\}$. Recall that a decoding path leading to clique C is defined by a particular combination of the clique states of all its parents. Thus the probability of occurrence of decoding path q can be written in terms of the probabilities of those clique states as

$$p_C^{(q)} = \prod_{C' \prec C, s_{C'} \in q} p_C^{(s_{C'})} = \prod_{l \in A_C^{(q)}} (1-\epsilon_l) \prod_{l \in \bar{A}_C^{(q)}} \epsilon_l \tag{3}$$

where $A_C^{(q)} = \bigcup_{C' \prec C, s_{C'} \in q} B_C^{(s_{C'})}$, and $\bar{A}_C^{(q)} = \bigcup_{C' \prec C, s_{C'} \in q} \bar{B}_C^{(s_{C'})}$. We now can write the expected distortion as a function of the transmission state

$$D(\boldsymbol{\epsilon}) = D_0 - \sum_C \sum_{q \in Q_C} p_C^{(q)} (\sum_{s \in S_C} p_C^{(s)} d_C^{(s,q)}) \tag{4}$$

where D_0 is the distortion of the GOPkt if no packets are decoded, $d_C^{(s,q)} = \sum_{l \in B_C^{(s)}} d_l^{(q)} - I_C^{(s,q)}$ directly derived from (1), and $p_C^{(q)}$ and $p_C^{(s)}$ are defined in (3) and (2) respectively. Theoretically the number of decoding paths may increase exponentially in the number of cliques preceding clique C. However, in practical pre-encoded applications, a number of decoding paths that lead to poor quality solutions can be discarded, or subsets of decoding paths that lead to the same reconstruction can be grouped together. This enables (4) to be tractable, as it can be simplified according to specific coding applications.

In the previous derivation of the expected distortion, we simply assume a one-to-one mapping between data units and packets. However, in general a data unit may be split into multiple packets as described in Section 3.2. In this scenario we express the PLP of data unit l, ϵ_l, as a function of the PLPs of its containing packets, ϵ_p, and use it in the equations (2) - (4). For the first case, when l can be decoded only when all its packets are received, it is straightforward to calculate ϵ_l as

$$\epsilon_l = 1 - \prod_{p \in l} (1-\epsilon_p). \tag{5}$$

ϵ_l is somewhat more complicated to express in the second case, when l is coded in an embedded stream. We assume that all the packets of l must be received in order for other data units, who depend on l for decoding, to be decodable. Therefore, ϵ_l in (3) to compute $p_C^{(q)}$ is calculated in the same way as (5). However, in terms of distortion, a subset of the packets in l can still achieve partial distortion reduction. Thus ϵ_l in (2), which is used again in (4), is given by

$$\epsilon_l = 1 - \sum_{p \in l} \rho_p \prod_{p' \preceq p} (1-\epsilon_{p'}) \tag{6}$$

In the ensuing discussions we simply assume one to one mapping between data units and packets unless it is explicitly noted. Extension to the second case can be easily achieved using (6), based on the previous discussion.

4. ANALYSIS BASED ON TAYLOR EXPANSION OF EXPECTED DISTORTION

The action of transmitting packets at each transmission time causes a state transition from one state $\boldsymbol{\epsilon_1}$ to another state $\boldsymbol{\epsilon_2}$. Since the expected distortion D is a function of $\boldsymbol{\epsilon}$, the state transition will change D correspondingly.

The goal of the scheduling algorithm is essentially to find an optimal transmission schedule to trigger a series of state transitions such that D at the final state is minimized given the channel conditions. Note that the ACK/NAK feedback from the receiver will cause state transitions as well. In this section, we apply the Taylor expansion to D in terms of $\boldsymbol{\epsilon}$ to reveal the characteristics of state transitions for different coding scenarios. This will be used to develop the scheduling algorithm proposed in Section 5.

A Taylor expansion of D at the current state $\tilde{\boldsymbol{\epsilon}}$ is given by

$$D(\boldsymbol{\epsilon}) = \sum_{k=0}^{\infty} \left[\frac{1}{k!} (\Delta\boldsymbol{\epsilon} \cdot \nabla_{\boldsymbol{\epsilon}'})^k D(\boldsymbol{\epsilon}') \right]_{\boldsymbol{\epsilon}'=\tilde{\boldsymbol{\epsilon}}} \tag{7}$$

$$= D(\tilde{\boldsymbol{\epsilon}}) + \sum_i a_i(\epsilon_i - \tilde{\epsilon}_i) + \sum_{i,j} a_{ij}(\epsilon_i - \tilde{\epsilon}_i)(\epsilon_j - \tilde{\epsilon}_j) + \dots \tag{8}$$

where $a_i = \frac{\partial D}{\partial \epsilon_i}|_{\epsilon_i=\tilde{\epsilon_i}}$ is the first-order partial derivative of D with respect to ϵ_i, $a_{ij} = \frac{\partial^2 D}{\partial \epsilon_i \partial \epsilon_j}|_{\epsilon_i=\tilde{\epsilon_i},\epsilon_j=\tilde{\epsilon_j}}$ is the second-order mixed partial derivative, and so on. Note that D depends linearly on $\boldsymbol{\epsilon}$, and (8) only contains the linear terms, i.e.

$$\frac{\partial^n D}{\partial \epsilon_{i_1}^{m_1}, \cdots, \partial \epsilon_{i_k}^{m_k}} = 0, \text{ if there exists a } m_j \geq 2 \text{ for } 1 \leq j \leq k.$$

a_i indicates the importance of packet i to the overall distortion at the current transmission state, based on the packet relationship in the group of packets (including dependency or redundancy) and the past transmission history of all related packets. However, it does not take into account the possibility of any future transmissions of other packets. The second or higher-order terms take effect when there are more than one packets whose PLPs have changed since the current state. They also show that a future change of the PLP of a packet through transmissions or receiving ACK/NAK may affect the current distortion benefit of its related packet. These effects show different characteristics depending on whether the coding applications allow only a single decoding path or multiple decoding paths.

4.1. Characteristics of Single Decoding Path

For coding applications with only a single decoding path, the expected end-to-end distortion can be simplified from (4) as

$$D(\boldsymbol{\epsilon}) = D_0 - \sum_l d_l \prod_{l' \preceq l} (1 - \epsilon_{l'}) \tag{9}$$

where $\prod_{l' \preceq l}(1 - \epsilon_{l'})$ is the probability that l is decodable. $l' \preceq l$ refers to the set of data units that must arrive at the receiver for l to be decoded. We can then derive its partial derivatives as

$$\frac{\partial D}{\partial \epsilon_i} = \sum_{l \succeq i} d_l \prod_{l' \preceq l, l' \neq i} (1 - \epsilon_{l'}) \tag{10}$$

$$\frac{\partial^2 D}{\partial \epsilon_i \partial \epsilon_j} = - \sum_{l \succeq i,j} d_l \prod_{l' \preceq l, l' \neq i,j} (1 - \epsilon_{l'}) \tag{11}$$

$$\dots$$

The higher-order derivatives can be derived similarly. It is easy to see that those partial derivatives lead to the following property:

Property 4.1. *$\frac{\partial^n D}{\partial \epsilon_{i_1}, \cdots, \partial \epsilon_{i_n}} \geq 0$ if n is odd, $\frac{\partial^n D}{\partial \epsilon_{i_1}, \cdots, \partial \epsilon_{i_n}} \leq 0$ if n is even.*

To understand this property, we consider (10) and (11) more closely. The right term in (10) can be written in two terms $f_1 + f_2$, where $f_1 = d_i \prod_{l' \prec i}(1 - \epsilon_{l'})$ shows the original distortion of packet i weighted by the probability of receiving all its parents, and $f_2 = \sum_{l \succ i} d_l \prod_{l' \preceq l, l' \neq i}(1 - \epsilon_{l'})$ indicates the importance of packet i to its children packets. $\frac{\partial D}{\partial \epsilon_i} \geq 0$ holds always, i.e., if packet i arrives it can only reduce the overall distortion. Similarly, the right term in (11) can also be written in two terms $-d_i \prod_{l' \prec i, l' \neq j}(1 - \epsilon_{l'})$ and $-\sum_{l \succ i, l \succeq j} d_l \prod_{l' \preceq l, l' \neq i,j}(1 - \epsilon_{l'})$.

These two terms together with the property $\frac{\partial D}{\partial \epsilon_i \partial \epsilon_j} \leq 0$ show that the importance of packet i increases when the probability of receiving its parent or children packets increases. In summary, Property 4.1 shows that, in the case of single decoding path, the arrival of one packet at the receiver can only increase, or at least not change the importance of the other packets.

4.2. Characteristics of Multiple Decoding Paths

For coding applications allowing multiple decoding paths, the expected end-to-end distortion is given in (4). We start by deriving its first-order derivative

$$\frac{\partial D}{\partial \epsilon_i} = f_1 + f_2 + f_3 + f_4, \text{ with} \tag{12}$$

$$f_1 = \sum_{q \in Q_C} p_C^{(q)} \Big[\sum_{s \in S_C, i \in B_C^{(s)}} d_C^{(s,q)} \prod_{l \in B_C^{(s)}, l \neq i} (1-\epsilon_l) \prod_{l \in \bar{B}_C^{(s)}} \epsilon_l \Big] \tag{13}$$

$$f_2 = -\sum_{q \in Q_C} p_C^{(q)} \Big[\sum_{s \in S_C, i \in \bar{B}_C^{(s)}} d_C^{(s,q)} \prod_{l \in B_C^{(s)}} (1-\epsilon_l) \prod_{l \in \bar{B}_C^{(s)}, l \neq i} \epsilon_l \Big] \tag{14}$$

$$f_3 = \sum_{C \succ C_i} \sum_{q \in Q_C, i \in A_C^{(q)}} \Big[\prod_{l \in A_C^{(q)}, l \neq i} (1-\epsilon_l) \prod_{l \in \bar{A}_C^{(q)}} \epsilon_l \Big] \cdot \Big[\sum_{s \in S_C} p_C^{(s)} d_C^{(s,q)} \Big] \tag{15}$$

$$f_4 = -\sum_{C \succ C_i} \sum_{q \in Q_C, i \in \bar{A}_C^{(q)}} \Big[\prod_{l \in A_C^{(q)}} (1-\epsilon_l) \prod_{l \in \bar{A}_C^{(q)}, l \neq i} \epsilon_l \Big] \cdot \Big[\sum_{s \in S_C} p_C^{(s)} d_C^{(s,q)} \Big] \tag{16}$$

where C_i represents the clique that contains packet i. The above four terms have different meanings: f_1 indicates the packet importance due to its own distortion reduction; f_2 shows the distortion effect of receiving i when its sibling packets have been received; f_3 shows the importance of i to its children cliques in the decoding paths which require i to be received; and f_4 presents the effect of receiving i to its children cliques in the remaining decoding paths which do not require i to be received. The signs of these terms indicate whether it is desirable to transmit i or not when different packets have been received at the decoder in the past. The sign of $\frac{\partial D}{\partial \epsilon_i}$ in this case cannot be easily derived directly from the above equations. However, in practical applications, receiving a packet will definitely not increase the overall distortion. If this were the case, one could choose not to decode this packet and D would not change. Therefore, $\frac{\partial D}{\partial \epsilon_i} \geq 0$ for any i.

It is more complicated to derive a general equation for the second-order mixed partial derivatives from (4). Instead, we look at a particular MDLC example shown in Fig. 3. For simplicity, we associate each packet BL_1, BL_2, EL_1, EL_2 and EL_0 with a tag from 1 to 5 in this order. Assume the current state $\tilde{\boldsymbol{\epsilon}} = [1, \cdots, 1]$ corresponds to the case where no packet has been sent. Consider two example second-order derivatives at $\tilde{\boldsymbol{\epsilon}}$.
(1) $\frac{\partial D}{\partial \epsilon_1 \partial \epsilon_2} = I_{C1}(1,1) > 0$, since BL_1 and BL_2 have redundancy with each other;
(2) $\frac{\partial D}{\partial \epsilon_1 \partial \epsilon_3} = -d_1^e < 0$, since BL_2 is dependent on BL_1 for decoding.
Different from that of single decoding path, $\frac{\partial D}{\partial \epsilon_i \partial \epsilon_j}$ does not have the same sign for any i and j. We can derive the same conclusion for the higher-order derivatives, which is omitted in the paper. To summarize, we have the following property:

Property 4.2. $\frac{\partial D}{\partial \epsilon_i} \geq 0$, $\frac{\partial^n D}{\partial \epsilon_{i_1}, \cdots, \partial \epsilon_{i_n}}$ $(n > 1)$ *can be either nonnegative or nonpositive.*

The property shows that, when there are multiple decoding paths, due to the redundancy between packets which affects the high-order terms, the future transmission of packets may decrease the current importance value of a packet that contains redundant information.

5. SCHEDULING ALGORITHMS WITH DAHG

5.1. Problem Formulation

The goal of scheduling is to minimize the playback distortion for a streaming session in a lossy packet network, by adapting to the network conditions and application requirements. Given the distortion expressed in (4), we could determine the optimal transmission policy using the Lagrangian optimized approach in [1]. However, the complexity of this approach increases greatly with the number of possible decoding paths. In this paper, we propose low-complexity scheduling techniques that are designed for cases when multiple decoding paths are possible, and where reduced complexity is important in order to make the decision among those multiple decoding scenarios at run-time.

Given a set of candidate packets G in the transmission time window, we define a schedule $\boldsymbol{\omega}$ as the transmission order of all these packets, which specifies whether and when to send each packet. We simply assume at each transmission time only one packet will be sent. Clearly, the overall redundancy of the selected subset of packets to be sent should match the channel behavior. The delivery order of packets has an important impact on the final playback distortion, due to the delay constraint and dependencies between packets. The scheduling problem can be stated as follows:

FORMULA 5.1. *Given a set of packets G and a channel model (channel bandwidth, packet loss rate, RTT and start-up delay), find the optimal schedule $\boldsymbol{\omega}^* \in \Omega$, such that the expected end-to-end distortion D is minimized, where Ω is the set containing all possible schedules under the rate constraints and delay requirements.*

Note that G will change over time, as packets which have been transmitted or expired (beyond the packet's time deadline) are removed, and new packets just becoming valid for transmission or those lost packets for retransmission are added. Therefore, $\boldsymbol{\omega}$ should be re-optimized after sending each packet to take into account the feedback information and the possible change of G since the previous transmission. That means we are only interested in the first packet in $\boldsymbol{\omega}$ at any given transmission time. This property was explored in [2, 6], and is used as well in our proposed low-complexity scheduling algorithms.

5.2. Greedy Algorithm

Since the first packet in $\boldsymbol{\omega}$ is used at any given time, instead of determining the complete transmission policy for each packet in G over all possible transmission opportunities (e.g. as used in [1, 5]), we use a greedy approach to select the currently most important packet from G to send. Our proposed greedy approach is based on *a prior* channel model, the past transmission history of the packets and the feedback from the receiver. Previous research work [2, 6] has proposed similar solutions for single-decoding-path codecs. Here, we derive the greedy algorithm for multiple-decoding-path codecs directly from the Taylor expansion of the expected distortion.

Let $\boldsymbol{\omega}_{i,0}$ be a transmission schedule such that packet i is not transmitted at the current time $\tilde{t}$, and let $\boldsymbol{\omega}_{i,1}$ be the same transmission schedule as $\boldsymbol{\omega}_{i,0}$ except that packet i will be transmitted at $\tilde{t}$. Sending packet i at $\tilde{t}$ induces a state transition from $\boldsymbol{\epsilon}(\boldsymbol{\omega}_{i,0})$ to $\boldsymbol{\epsilon}(\boldsymbol{\omega}_{i,1})$, and thus leads to a distortion reduction by

$$\Delta D_i^{(\tilde{t})} = D(\boldsymbol{\epsilon}(\boldsymbol{\omega}_{i,0})) - D(\boldsymbol{\epsilon}(\boldsymbol{\omega}_{i,1})) = a_i(\epsilon_{i,0} - \epsilon_{i,1}) \tag{17}$$

derived from (8), where $\epsilon_{i,0}$ and $\epsilon_{i,1}$ are the PLP of packet i given the schedule $\boldsymbol{\omega}_{i,0}$ or $\boldsymbol{\omega}_{i,1}$, respectively. $\Delta D_i^{(\tilde{t})}$ indicates the importance of sending packet i at the current time $\tilde{t}$. Assume that a packet will be sent only if it has not been sent or if a NAK from the receiver has arrived. If no further transmission of packet i is considered, the right hand side of (17) becomes $a_i(1-\epsilon)$, where ϵ is the current channel packet loss rate. However there could be possible retransmissions for packet i before its time deadline t_i. To take into account the effects of different deadlines for each frame as in [2], we approximate $\epsilon_{i,0}$ as ϵ^{m_i}, where m_i is the number of possible retransmissions given by

$$m_i = (t_i - \tilde{t})/RTT. \tag{18}$$

$\epsilon_{i,1}$ then becomes ϵ^{m_i+1}. Ignoring the constant term $(1-\epsilon)$ and taking into account the packet size r_i, we have the metric

$$c_i = \epsilon^{m_i} \frac{a_i}{r_i} \tag{19}$$

for each packet and select the one with the largest c_i to send. Note that a_i is calculated at the current state with the assumption that there are no future transmissions of other packets. Table 1 summarizes the algorithm.

Algorithm 1 (Greedy algorithm)

1. compute (19) for each packet i in G
2. Find the largest c_i, say j (i.e. $c_j \geq c_i$ for any $i \neq j$)
3. send packet j

Table 1. Greedy algorithm

5.3. M-T Algorithm

The main problem for the greedy algorithm is that it does not take into account the possibility of any future transmissions of other packets. As Property 4.2 points out that, for applications with multiple decoding paths, the future transmission of a packet may either increase or decrease the importance value of another packet depending on their coding relation. Thus, the further transmission probabilities of packets have increased effects, through the high-order terms of the Taylor expansion, on the decision at the current transmission opportunity if an optimal scheduling algorithm is used. In this section, we propose to use the M-algorithm [7] with a look-ahead window T, which we simply called the M-T algorithm. This algorithm can be regarded as an intermediate step between the greedy algorithm, which only considers the local time step, and the Lagrangian optimal algorithm, which optimizes the complete transmission policy for all the future transmission opportunities. In addition to the trade-off consideration between algorithm performance and complexity, we also avoid to make the transmission decisions for the future time too far away, because the information used to make decisions may change along the time (such as the group of packets G and feedback information), and therefore the decisions may change as well.

In the M-T algorithm, we first introduce a look-ahead time window T to take into account the possible future transmissions, and implicitly account for the high-order terms in the expansion. T is usually small and that means we only consider the effects of the possible transmissions in the near future on the current decision. When the GOPkt is relatively large, it is intractable to perform an exhaustive search for the best schedule even over a short time window T. Thus, we adopt the M-algorithm which is initially developed as a sub-optimal search technique through a trellis. At each step, only M best schedules are retained as survivors and carried over to next step. Starting from these states in the next step, we repeat the same search process and again only the best M are retained as survivors, while the rest are eliminated. This process repeats over the time window until time T, where we select the best one as our schedule.

Now we define the cumulative metric used in the algorithm for schedule selection. Without loss of generality, we label the current time step as time 0. Let $\boldsymbol{\omega} = [\omega_0, \cdots, \omega_{T-1}]$ be a length-T vector to represent a candidate transmission schedule of the look-ahead window, and ω_t is the index of the packet that is to be sent at time t under the schedule $\boldsymbol{\omega}$. Let $\boldsymbol{\epsilon}^{(t)}(\boldsymbol{\omega})$ be the transmission state at time t for a given $\boldsymbol{\omega}$, and $\epsilon_i^{(t)}(\boldsymbol{\omega})$ be the PLP of packet i at state $\boldsymbol{\epsilon}^{(t)}(\boldsymbol{\omega})$. (17) gives the distortion reduction of sending packet i at a single time step. Then the cumulative distortion reduction in the period of time t is given by

$$\Delta D(\boldsymbol{\omega}, t) = \sum_{\tau=0}^{t-1} \Delta D_{\omega_\tau}^{(\tau)}. \tag{20}$$

Similar to the metric used in the greedy algorithm, we can define the cumulative metric at time t as

$$c_{\boldsymbol{\omega}}(t) = \Big[\sum_{\tau=0}^{t-1} \epsilon^{m_{\omega_\tau}} a_{\omega_\tau}^{(\tau)}\Big] \Big/ \sum_{\tau=0}^{t-1} r_{\omega_\tau} \tag{21}$$

Where $a_{\omega_\tau}^{(\tau)}$ is the first-order partial derivative of D with respect to packet ω_τ at state $\boldsymbol{\epsilon}^{(\tau)}(\boldsymbol{\omega})$. Table 2 summarizes the algorithm.

6. EXPERIMENTAL RESULTS

We evaluate the rate-distortion performance for different scheduling algorithms based on the MDLC example. In this simulation, we first generate an i.i.d. Gaussian sequence z_k with mean zero and variance one. We

Algorithm 2 (M-T algorithm)

1. Initialize: $\tilde{G} = G$
2. compute (19) for each packet i in $\tilde{G}$
3. choose the set of packets $\{l_1, \cdots, l_M\}$ with the M largest c_i
4. update $\Omega^{(0)} = \{\boldsymbol{\omega}_1^{(0)}, \cdots, \boldsymbol{\omega}_M^{(0)}\}$, $\boldsymbol{\omega}_i^{(0)} = [l_i, 0, \cdots, 0]_{T\times 1}$
5. **for** $t = 1$ to $T-1$
6.   **for** $i = 1$ to M
7.   update $\tilde{G}$ by removing packets having been expired
8.   compute (21) for each packet j in $\tilde{G}$, where $\boldsymbol{\omega} = [\boldsymbol{\omega}_i^{(t-1)}(0, \cdots, t-1), j, 0, \cdots, 0]$
9.   choose the M largest $c_{\boldsymbol{\omega}}(t)$ as candidate schedule set $\Omega_i^{(t)}$
10.   **end**
11.   update $\Omega^{(t)}$ with the M largest $c_{\boldsymbol{\omega}}(t)$ from candidate schedule sets $\Omega_1^{(t)}, \cdots, \Omega_M^{(t)}$
12. **end**
13. $\boldsymbol{\omega}^* = \text{argmax}_{\boldsymbol{\omega}\in\Omega^{(T-1)}} \ [c_{\boldsymbol{\omega}}(T-1)]$
14. send the first packet in $\boldsymbol{\omega}^*$

Table 2. M-T algorithm

split the sequence into even and odd sequences, quantize them with both fine and coarse quantizers, and then pair finely quantized even samples and coarsely quantized odd samples, or coarsely quantized even samples and finely quantized odd samples into two base layer descriptions, respectively. Each base layer description of a pair $[z(2k), z(2k+1)]$ is put into one packet, and the two samples are considered as a single unit with the same delivery deadline. Three enhancement layer descriptions are created as Fig. 1 with a finer quantization step. We use the Lloyd-max algorithm to optimize the quantization parameter, and quantize the sequence. The entropy of the quantized symbol is used to approximate the rate required to code the quantized sequence. The rate for each BL_1/BL_2 is 4.9346, while the rate for EL_0 is 4.6802, and EL_1/EL_2 6.1642. The distortion related quantities are assigned in the same way as the example shown in Fig. 3, with the values given in table 3.

Table 3. Distortion values for Gaussian coded sequence (symbol notation shown in Fig. 3).

d_0^b	d_1^b	d_0^e	d_1^e
0.98781	0.81168	0.01215	0.18828

Fig. 4 shows the SNR of the reconstructed sequence with various channel bandwidths when different scheduling algorithms are used. The Lagrangian approach [1] performs best especially in the low-rate range. The M-T algorithm outperforms the greedy algorithm between 0.5 to 2 dB. The main problem for the Lagrangian approach is its high computational complexity that grows exponentially in the number of transmission opportunities N. The performance of this approach drops quickly with a smaller N. Here we show the experimental results at $N = 12$, which is among the best performance we got in our implementation. The Lagrange multiplier λ is fixed for one run of the complete sequence. The average rate increases as λ decreases. However, the number of packets selected at each transmission opportunity may vary, resulting in a variable instant transmission rate during the streaming. For L packets in a GOPkt, the complexity for greedy algorithm is $O(L)$, while that of the M-T algorithm is $O(LMT)$. Fig. 5 compares the performance of greedy algorithm and M-T algorithm under various channel packet loss rates. Since the given channel bandwidth is more than enough to transmit any single description to the decoder in an error-free channel, both algorithms achieve the highest SNR when $\epsilon = 0$. As ϵ increases M-T algorithm performs much better than greedy algorithm. And when the channel becomes even worse at about 30% loss of the time, these two algorithms behave closely.

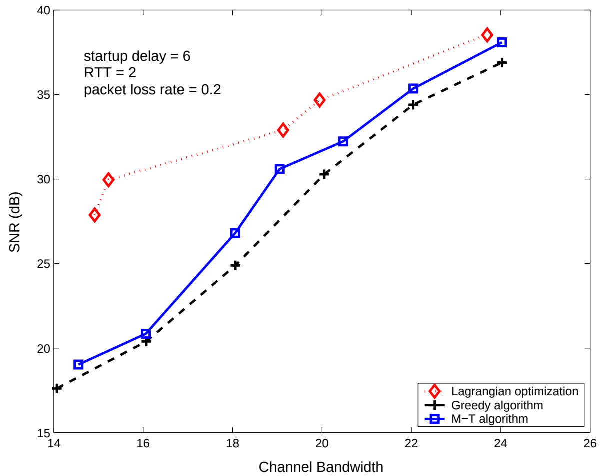

Figure 4. The comparison of SNR under different scheduling algorithms with various channel bandwidths. (Lagrangian algorithm: $N = 12$. M-T algorithm: $M = 8$, $T = 8$.)

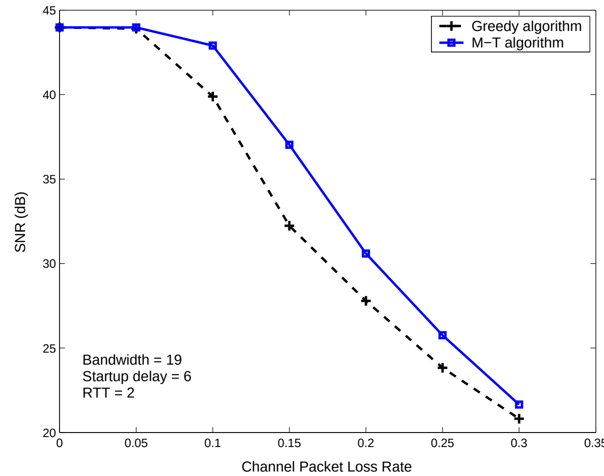

Figure 5. The comparison of SNR under different scheduling algorithms with various channel packet loss rates. (M-T algorithm: $M = 8$, $T = 8$)

7. CONCLUSIONS

In this paper we have extended recent work on rate-distortion based video scheduling to the general case where multiple decoding paths are possible. We proposed a new source model called Directed Acyclic Hypergraph (DAHG) to describe the decoding dependence and redundancy between different data units. Based on this model, we have proposed two low-complexity scheduling algorithms, i.e., the greedy algorithm and the M-T algorithm. The preliminary results show that the M-T algorithm outperforms the greedy algorithm by taking into account some of the possible future transmissions.

REFERENCES

1. P. A. Chou and Z. Miao, “Rate-distortion optimized streaming of packetized media,” *submitted to IEEE Trans. Multimedia* , Feb. 2001.
2. Z. Miao and A. Ortega, “Expected run-time distortion based scheduling for delivery of scalable media,” in *12th International Packet Video Workshop (PVW 2002)*, Apr. 2002.
3. G. Cheung and W. Tan, “Directed acyclic graph based source modeling for data unit selection of streaming media over QoS networks,” in *IEEE Int. Conf. Multimedia and Expo (ICME 2002)*, Aug. 2002.
4. H. Wang and A. Ortega, “Robust video communication by combining scalability and multiple description coding techniques,” in *SPIE Conf. Image and Video Commun. and Processing (EI 2003)*, Jan. 2003.
5. M. Kalman and B. Girod, “Rate-distortion optimized streaming of video with multiple independent encodings,” in *Proc. IEEE Int. Conf. Image Processing (ICIP 2004)*, Oct. 2004.
6. Z. Miao and A. Ortega, “Fast adaptive media scheduling based on expected run-time distortion,” in *Proc. Asilomar Conf. Signals, Systems and Computers*, Nov. 2002.
7. J. B. Anderson and S. Mohan, “Sequential coding algorithm: A survey and cost analysis,” *IEEE Trans. Commun.* **32**, pp. 169–176, Feb. 1984.