EFFICIENT FRAME TRANSMISSION FOR SCALABLE VIDEO STREAMING WITH DEPENDENCY STRUCTURE

by

Saied Mehdian

A thesis submitted in conformity with the requirements for the degree of Master of Applied Science
Graduate Department of Electrical and Computer Engineering
University of Toronto

© Copyright 2012 by Saied Mehdian
Abstract

Efficient Frame Transmission for Scalable Video Streaming with Dependency Structure

Saied Mehdian
Master of Applied Science
Graduate Department of Electrical and Computer Engineering
University of Toronto
2012

Efficient transmission schemes are presented for streaming scalable video over a link with limited capacity. The objective is to select a transmission sequence of data units and their transmission schedule such that the overall video quality is maximized. For video with a single spatial layer, optimal transmission schemes are obtained for two general classes of hierarchical prediction structures, which include the popular dyadic structure. Based on a new characterization of the interdependence among frames in terms of trees, structural properties of an optimal transmission schedule are derived. These properties lead to the development of a jointly optimal frame selection and scheduling algorithm, which has computational complexity that is quadratic in the number of frames. Then, using the concept of virtual deadlines, an efficient sub-optimal scheme for the transmission of video with multiple spatial layers is proposed. Simulation results show that the proposed schemes substantially outperform two existing alternatives.
Dedication

To my parents and my sister
Acknowledgements

I would like to express my sincere gratitude to my advisor, Professor Ben Liang. Words can not express how grateful I am for his generous guidance, patience, support and encouragement during the past two years. During this time, I have learnt alot from Ben; one of the things I admire the most is him teaching me how to be a self-motivated and independent researcher. I also would like to thank him for introducing me to the exciting and challenging research topic of video streaming. The research experience under his supervision has given me the opportunity to better understand my potentials and incorporate my abilities to pursue my goals through my career.

I would like to thank my oral examination committee members, Professor Ashish Khisti, Professor Baochun Li and Professor Peter R. Herman for reading my thesis and providing me with their invaluable comments and feedbacks.

I would like to thank my parents, Mohammadhossein and Nahid, and my sister, Mina, for their endless encouragement and support throughout my life.

I acknowledge the financial support of the research for this thesis by Bell Canada, Natural Sciences and Engineering Research Council (NSERC) Collaborative Research and Development grant and NSERC Discovery grant. Also, I acknowledge the Edward S. Rogers Sr. Scholarship provided by University of Toronto.
Contents

1 Introduction .................................................. 1
  1.1 Overview .................................................. 1
  1.2 Thesis Organization ....................................... 3

2 Background and Related Work ............................. 4
  2.1 Motion Prediction ......................................... 4
  2.2 Related Work ............................................. 4
    2.2.1 Classical Deterministic Scheduling ..................... 4
    2.2.2 Scheduling for Scalable Video Streaming ............... 6

3 Hierarchical Prediction Structure and Problem Statement ... 7
  3.1 Hierarchical Prediction Structure ....................... 7
  3.2 Quality Optimization under Limited Link Capacity ........... 8

4 Properties of Optimal Schedule ............................ 10
  4.1 Modified Breadth First Search (MBFS) Trees ................. 10
  4.2 SIO and Quasi-SIO Classes ................................ 11
  4.3 Canonical Form of Optimal Transmission Sequence .......... 12

5 Optimal Schedule for the SIO Class ....................... 14
  5.1 SIO Universal Sequence .................................... 14
  5.2 Computation of Optimal Schedule ......................... 15

6 Optimal Schedule for the Quasi-SIO Class .................. 17
  6.1 Quasi-SIO Universal Sequence ............................ 17
  6.2 Computation of Optimal Schedule ......................... 20

7 Multi-layer Transmission Extension ......................... 22
  7.1 Simultaneous Video Transmission ......................... 22
    7.1.1 The Cousin Relationship .............................. 22
  7.2 Analysis of Multi-layer SVC transmission ................... 25
    7.2.1 Step 1: Layer-wise Sequencing ......................... 26
    7.2.2 Step 2: Resource Extraction and Deadline Update ....... 26
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>Experiment with Video Traces</td>
<td>29</td>
</tr>
<tr>
<td>8.1</td>
<td>Single Layer Transmission Simulation</td>
<td>29</td>
</tr>
<tr>
<td>8.1.1</td>
<td>Methodology</td>
<td>29</td>
</tr>
<tr>
<td>8.1.2</td>
<td>Experimental Results</td>
<td>29</td>
</tr>
<tr>
<td>8.2</td>
<td>Multi-layer Transmission Simulation</td>
<td>31</td>
</tr>
<tr>
<td>8.2.1</td>
<td>Methodology</td>
<td>31</td>
</tr>
<tr>
<td>8.2.2</td>
<td>Experimental Results</td>
<td>31</td>
</tr>
<tr>
<td>9</td>
<td>Conclusions, Extensions and Future Work</td>
<td>33</td>
</tr>
<tr>
<td>9.1</td>
<td>Conclusions</td>
<td>33</td>
</tr>
<tr>
<td>9.2</td>
<td>Extensions</td>
<td>33</td>
</tr>
<tr>
<td>9.3</td>
<td>Future Work</td>
<td>34</td>
</tr>
<tr>
<td>10</td>
<td>Appendices</td>
<td>35</td>
</tr>
<tr>
<td>10.1</td>
<td>Proof of theorem 3.1</td>
<td>35</td>
</tr>
<tr>
<td>10.2</td>
<td>Proof of Lemma 4.1</td>
<td>36</td>
</tr>
<tr>
<td>10.3</td>
<td>Hierarchical Dyadic Structure</td>
<td>36</td>
</tr>
<tr>
<td>10.4</td>
<td>Proof of Theorem 4.1</td>
<td>41</td>
</tr>
<tr>
<td>10.5</td>
<td>Proof of Lemma 4.2</td>
<td>42</td>
</tr>
<tr>
<td>10.6</td>
<td>Proof of Lemma 4.3</td>
<td>42</td>
</tr>
<tr>
<td>10.7</td>
<td>Proof of Lemma 5.2</td>
<td>43</td>
</tr>
<tr>
<td>10.8</td>
<td>Proof of Lemma 6.5</td>
<td>43</td>
</tr>
<tr>
<td>10.9</td>
<td>Proof of Lemma 6.6</td>
<td>43</td>
</tr>
<tr>
<td>10.10</td>
<td>Proof of Theorem 7.1</td>
<td>45</td>
</tr>
<tr>
<td>10.11</td>
<td>Proof of Lemma 7.1</td>
<td>45</td>
</tr>
<tr>
<td>10.12</td>
<td>Proof of Lemma 7.2</td>
<td>45</td>
</tr>
<tr>
<td>10.13</td>
<td>Proof of Lemma 7.3</td>
<td>46</td>
</tr>
<tr>
<td>10.14</td>
<td>Proof of Lemma 7.4</td>
<td>46</td>
</tr>
<tr>
<td></td>
<td>Bibliography</td>
<td>47</td>
</tr>
</tbody>
</table>
List of Figures

2.1 Examples of Hierarchical Prediction Structures. ........................................ 5
3.1 Frames (0, 1, . . . , 16) form a GOP with hierarchical dyadic structure G16B3. .... 8
4.1 MBFS on the DAG in Figure 3.1 with node 16 removed. .......................... 12
6.1 Critical nodes in the GOP, MBFS tree of the quasi-SIO class, plus edges from \( I_{i+1} \). Each triangle indicates a subtree rooted at the node located at its top. ............. 19
7.1 (a) A schedule in which \( l_2 \) is placed prior to \( l_1 \) but \( l_1 \rightarrow l_2 \). (b) Transformation to meet the first property. ................................................................. 24
7.2 Transformation corresponding to the second property in Theorem 4.1. ............ 25
7.3 Dependence graph of a multi-layered scalable video. .................................. 25
7.4 Construction of the time horizon referring to the scheduling of the \( k \)-th layer given preceding layers are already scheduled. ................................. 27
8.1 Y-PSNR vs. link capacity \( C \) for single spatial layer diagrams. .................. 30
8.2 Y-PSNR vs. link capacity \( C \) for multi spatial layer diagrams. ..................... 32
10.1 Transforming a preemptive schedule to a non-preemptive schedule. .......... 35
10.2 Transforming simultaneous transmissions to nonsimultaneous transmissions. ........ 36
10.3 Hierarchical dyadic structure with \( \omega = 1 \) used in the proof of Theorem 10.1. \( x \) and \( z \) are non-B-frames, and \( y \) is a B-frame. ................................................................. 37
10.4 Hierarchical dyadic structure with \( \omega = k \) used in the proof of Theorem 10.1. \( x \) and \( z \) are non-B-frames, and all frames between them are B-frames. ......................... 38
10.5 Hierarchical dyadic structure with \( \omega = k + 1 \) used in the proof of Theorem 10.1. \( x \) and \( w \) are non-B-frames, and all frames between them are B-frames. ....................... 39
10.6 (a) A schedule in which \( l_2 \) is placed prior to \( l_1 \) but \( l_1 \rightarrow l_2 \). (b) Transformation to meet the first property. ................................................................. 41
10.7 Transformation corresponding to the second property in Theorem 4.1. ........... 42
List of Acronyms

AVC   Advanced Video Coding
CGS   Coarse Grain Scalability
DAG   Directed Acyclic Graph
DPQ   Delta Quantization Parameter
DU    Data Unit
EDF   Earliest Deadline First
GOP   Group of Pictures
HPS   Hierarchical Prediction Structure
MBFS  Modified Breadth First Search
PBEDF Priority Based Earliest Deadline First
Quasi-SIO Quasi Sequential and Isomorphically Ordered
SIO   Sequential and Isomorphically Ordered
SVC   Scalable Video Coding
Chapter 1

Introduction

1.1 Overview

Video applications currently serve as one of the most important parts of online services driving the increasing demand for network resources. In 2016, internet video traffic is predicted to comprise 55% of all consumer Internet traffic while this amount was 51% in 2011 [1].

One of the most significant video applications is video streaming. In January 2012, Youtube, one of the biggest video streaming service providers, reported it surpassed 4 billion views per day [2]. Video streaming corresponds to downloading and playing back a video clip simultaneously. A video clip is simply a sequence of frames which is displayed with a constant rate. In video streaming, at the user end, while some parts of the video are displayed, some other parts of the video are downloaded from a media server. This application is in contrast to complete downloading and then playing back, where the user has to wait for the entire video clip to be downloaded. Video streaming is not confined to stored video services; video conferencing and live broadcasting are applications which become possible only through video streaming.

The main challenge of video streaming, which makes it also fundamentally different from file downloading, is the delay requirements. In particular, during video playback, once a frame is displayed, the user expects to see the next frames within a regular time interval. The failure of on-time delivery of the data of next frames results in degradation of video quality or even a display freeze, both of which are distracting for the audience. The problem of delay originates from the fact that video data have relatively high source rates compared to the available network bandwidths. Furthermore, the situation is exacerbated since network resources are shared among multiple users.

However, video streaming is contributing to an ever increasing portion of the Internet traffic going through service providers. Beside stringent demands on bandwidth and delay, it requires adaptation to heterogeneous access networks and devices, in order to achieve satisfactory viewing experience. The efficient and adaptive transmission of video is paramount to service providers and users alike.

Toward this goal, one thrust of engineering effort is on efficient video coding. The H.264 Advanced Video Coding (AVC) standard, including its Scalable Video Coding (SVC) enhancements [3][4], is the most widely adopted coding scheme, used in a wide array of applications including Bluray and Youtube. It provides a means to encode a video stream into sub-streams of different quality, which may be selectively transmitted and decoded based on the available communication hardware and bandwidth.
However, the H.264 standard does not specify how to select or schedule video data for transmission. Although it is possible to dynamically encode a video based on the available transmission rate, the available channel rates in practice are very unlikely to satisfy video streaming applications, especially when multiple users try to share the same transmission media. Hence, exploiting the transmission resources in order to boost the quality of service is a necessity.

The other engineering thrust is on adaptive transmission schemes for scalable video streaming. It concerns the joint optimization of two procedures, first, the selection of a subset of the video frames for transmission, and second, scheduling the transmission of those frames before their display deadlines expire. This is challenging mainly due to the *Hierarchical Prediction Structure (HPS)* of the video codec [4]. Prediction is necessary to improve coding efficiency, but it creates complicated dependencies between frames in a video sequence. More details on prediction will be provided later in Section 2.1. Most existing studies on efficient transmission of scalable video [5]-[12] address dependence by a simplified flow-based model, in which a video layer can be decoded only if all lower layers are already decoded. In the frame-based approach, dynamic frame dropping and retransmission have been studied in [13]-[19], but these proposals are mostly based on heuristics that are challenging to analyze mathematically.

In this work, for the transmission of SVC video with at single spatial layer, we present an analytical approach to optimize the frame transmission schedule of prestored videos with hierarchical prediction, with joint consideration for the optimal frame selection given a limited link capacity. The optimization objective is to minimize the loss of playback quality in the transmitted video sequence. Note that in this thesis, dynamic encoding of the video is not considered; at least for one reason, it will be computationally intensive and time consuming. However, if the online encoding delays are known, by only adjusting the deadlines of the frames, the schemes proposed in this thesis still work; therefore, it might be possible to develop a joint source coding and channel scheduling scheme. To the best of our knowledge, this is the first work to provide a provably optimal solution to frame-by-frame lossy video transmission which accounts for hierarchical prediction. Furthermore, for SVC multi-spatial layer prestored video transmission, a sub-optimal solution is proposed by giving priorities to the layers and scheduling one spatial layer at a time.

Our main contributions are as follows.

- **First**, we formally characterize the inherent properties of an optimal frame transmission schedule in terms of the dependency structure arising from the HPS. This leads to the categorization of two general classes of videos, termed Sequential Isomorphically Ordered and Quasi Sequential Isomorphically Ordered, which include as special cases some commonly employed prediction schemes such as the hierarchical dyadic structure [4].

- **Second**, we develop efficient algorithms to compute the frame ordering rules in an optimal transmission schedule. We show that they hold regardless of the subset of frames selected for transmission and, in particular, governs a unique universal schedule of which an optimal schedule is a subsequence.

- **Third**, based on the above structural observations, we propose dynamic programming solutions for jointly optimal frame selection and scheduling, which is shown to have only quadratic complexity. Simulation with common test video traces show that the proposed method can substantially improve the quality of lossy video streaming over a link with limited capacity.
• *Forth*, by presenting the concept of virtual deadlines and using the single layer transmission scheme, a suboptimal scheme for the transmission of multi spatial layer SVC is proposed.

### 1.2 Thesis Organization

The rest of this thesis is organized as follows. In Chapter 2, some backgrounds on video coding and related works on scheduling are presented. In Chapter 3, the system model and problem statement are explained. In Chapter 4, the special properties of an optimal schedule are discussed. In Chapters 5 and 6, optimal algorithms for scheduling in the two general classes of HPS are presented. In Chapter 7, an efficient algorithm for multi spatial layer video transmission is proposed. In Chapter 8, simulation results with video traces are shown to demonstrate the performance gain by the proposed solutions. Finally, Chapter 9 concludes the thesis.
Chapter 2

Background and Related Work

2.1 Motion Prediction

As mentioned in the previous chapter, video comprises of a sequence of frames. Storing individual frames in order to maintain a video sequence results in extremely large video sizes and rates. Frame compression techniques alone is insufficient.

However, consecutive video frames usually contain data that are substantially correlated. Video compression techniques intend to exploit these redundancies combined with frame compression techniques to reduce the overall video size and rate. These redundancies are exploited using the Motion Prediction or Motion Compensation technique. This technique is based on the fact that two close frames in a video sequence are similar except that some parts may have changed due to motion. Motion Prediction tries to build up one of the frames by predicting and applying these motions to the other frame.

In the H.264 AVC standard, in order to implement motion prediction, all frames are divided into indexed blocks. In order to construct a frame, the blocks of the reference frame are moved around by motion vectors; the motion vectors are specific to each block to be built and the reference frame.

After the motion vectors are applied, the difference between the actual block and the moved block, residual signal, is recorded using transformation techniques such as wavelet and discrete cosine transform and then quantized. The motion vectors are intended to be chosen such that the residual signal power is minimized.

As a result, the decoding of some frames will depend on the availability of other frames. Some of the popular prediction structures are shown in Figure 2.1. In this thesis, the dependency structure among frames is exploited to design efficient video transmission schemes.

2.2 Related Work

Previous works related to this paper can be categorized into two groups: classical scheduling and scalable video streaming.

2.2.1 Classical Deterministic Scheduling

Classical scheduling problems related to our work are addressed under the theory of deterministic scheduling with delay constrained jobs [20], [21]. In these problems as set of jobs are to be processed through
Figure 2.1: Examples of Hierarchical Prediction Structures.
Chapter 2. Background and Related Work

a machine. Each job has a nonzero processing time and the objective is to minimize a weighted sum of late jobs. In [22] and [23], scheduling schemes for independent jobs are presented. Nevertheless, the schedule may need to satisfy some precedence relation between jobs. However, if a job is discarded, it has no effect on the other jobs [24]. This is different from the dependency relation between video frames. To the best of our knowledge, there is no known general theory on scheduling with dependency.

2.2.2 Scheduling for Scalable Video Streaming

Many prior works do not consider the dependency relation in video transmission [25]-[32]. In all these works the video content is transmitted in an earliest-deadline-first (EDF) fashion. In [27], a dropping scheme is proposed which does not take into account deadline constraints. In particular, for each regular time interval, a constant size set of consecutive frames is considered for transmission and some of them are dropped to comply with channel rate limit, regardless of deadline constraints. In [25], [26], a dropping scheme utilizing dynamic programming is presented which dynamically adjusts the transmission power and therefore the rate to ensure selected packets meet their deadlines. Studies that do consider the dependency relation can be divided into two main groups: flow based and frame based. In the flow-based approach [5]-[12], the video is modelled as a set of inter-dependent data flows, each providing basic or enhanced playback quality. Flow-based dependency structure often is relevant only within the same display frame. In this work, we consider the more complicated frame-based dependency structures.

In the frame-based approach, the video sequence is modelled as a set of data units, roughly corresponding to the display frames. Each data unit may depend on one or more data units for decoding. In [13]-[16], heuristics are proposed to drop data units under bad link conditions according to some pre-defined priority. No analytical result is presented on how to set priority levels, and evaluation is performed through simulation only. In comparison, our proposed scheduling algorithm for single spatial layer transmission is provably optimal. However, our proposed algorithm for multi spatial layer transmission is suboptimal.

More sophisticated online frame-based retransmission schemes have been studied for the lossy environment in [17]-[19]. In [17], a heuristic retransmission windowing approach is used to improve the rate-distortion performance of video delivery over an erasure channel. In addition, [17] intends to minimize a cost function including as a linear term the amount of transmitted data during an entire video streaming session, whereas in this thesis, a channel rate limit is considered which not only restricts the amount of transmission data during the entire session but also at each time instant, which is a much more realistic constraint. In [18], a content-aware hybrid ARQ method for video streaming over wireless links is proposed and analysis using Markov chains, where frames are transmitted in EDF fashion and retransmission are used for I-frames. In [19], an approximate analytical model for multi-user video transmission over wireless links is developed, for videos that do not use B-frames. These works target more ambitious problems than ours, obtaining weaker suboptimal solutions. In this work, we target a simpler offline decision problem with an optimal scheduling algorithm.
Chapter 3

Hierarchical Prediction Structure and Problem Statement

We consider a time-slotted point-to-point communication scenario where the transmitter sends a pre-recorded video sequence to the receiver through a link with fixed capacity $C$ bits/timeslot. We assume that a lower-layer protocol ensures the correct reception of any data transmitted at or below the link capacity. Without loss of generality, we omit the propagation delay on the link, since otherwise we only need to shift the display time of all frames by a constant offset to accommodate it.

3.1 Hierarchical Prediction Structure

The video consists of a sequence of frames indexed in their display order by 1, 2, \ldots, $N$. The frames are classified into three groups: I-frames, P-frames, and B-frames. I-frames are intra-coded and do not depend on other frames, P-frames are inter-coded based on a preceding frame in the display order, and B-frames are inter-coded based on a preceding frame and a successive frame. The set of P-frames and B-frames between two consecutive I-frames, plus the leading I-frame, is called a group of pictures (GOP). The P-frames and B-frames of different GOPs are isolated from each other, but those within a GOP adhere to a specific dependency structure governed by the adopted prediction coding. This corresponds to the general HPS of H.264 AVC, which enables temporal scalability [4]. However, different GOPs can have different structures.

As an example, the most widely adopted HPS is the hierarchical dyadic structure. Following the notations of [33], a hierarchical dyadic structure is denoted by $GnBm$ where $n$ is the size of each GOP and $m$ is the number of B-frames between consecutive I-frames or P-frames, with $m = 2^\omega - 1$ for some $\omega \in \mathbb{N}$. Figure 3.1 illustrates a GOP with $G16B3$.

The dependency between frames is represented as a directed acyclic graph (DAG), where frames are nodes, and their dependencies are indicated by edges [17], as shown in Figure 3.1. If decoding frame $l$ requires frame $l'$ directly, which is denoted by $l' \rightarrow l$, then a direct edge connects node $l'$ to node $l$, and $l'$ is called a parent of $l$, and $l$ a child of $l'$. If the decoding of frame $l$ depends on frame $l'$, possibly through some intermediate nodes, then $l'$ is called an ancestor of $l$, and $l$ a descendant of $l'$. If two frames have no ancestral relation, they are called irrelevant and denoted by $l \not\rightarrow l'$. Throughout this paper, we use $G$ to denote the DAG corresponding to the video sequence under consideration, and the terms “frame”
Chapter 3. Hierarchical Prediction Structure and Problem Statement

Figure 3.1: Frames (0, 1, ..., 16) form a GOP with hierarchical dyadic structure $G16B3$.

and “node” are used interchangeably depending on context.

We denote the sequence of GOPs as $\{GOP_i\}$ in their display order. We also denote the sequence of I-frames as $\{I_i\}$ in their display order, so that $I_i$ corresponds to $GOP_i$. The descendants of $I_{i+1}$ can only be in $GOP_i$ and $GOP_{i+1}$, and except $I_{i+1}$, the frames in $GOP_i$ cannot depend on any frame in other GOPs. Let $\mathcal{N}_i$ be the set of non-I-frames in $GOP_i$, and $\mathcal{D}_i$ be the set of frames in $GOP_i$ that are descendants of both $I_i$ and $I_{i+1}$.

3.2 Quality Optimization under Limited Link Capacity

Each video frame $l$ is associated with three parameters $(S_l, d_l, q_l)$, where $S_l$ is its size in bits, $d_l$ is its display deadline offset against some given transmission start time (i.e., time 0), and $q_l$ is its quality increment, the expected loss of video quality if the frame is not displayed. For example, one possible measure of $q_l$ is the peak signal-to-noise ratio. Denote the deadline of the last frame as $\hat{T}$.

A transmission schedule is a vector containing the transmission starting times of a sequence of video frames, sorted in ascending order. It indicates both the selected frames for transmission and the order of transmission. In a transmission schedule, a frame is decodable if and only if its parents are already decoded. A frame is successful if it becomes decodable and arrives at the receiver prior to its display deadline.

The reward function of a transmission schedule, $\mathcal{S}$, is defined as the sum of the quality increment of its successful frames, i.e.,

$$z(\mathcal{S}) = \sum_{l \in \mathcal{S}, l \text{ is successful}} q_l$$

Our objective is to find a transmission schedule that maximizes the reward given a link capacity limit $C$. This problem is called the Scheduling Problem throughout the rest of this paper. The resultant transmission schedule is called an optimal schedule, and the resultant reward function is indicated with $z^*$.

Note that an optimal schedule may permit unsuccessful frames to be transmitted. Although these frames are not displayed, their importance arises from the fact that due to the dependency structure, they can help decode other frames. However, the transmission of an unsuccessful frame which has no successful descendant is useless. Therefore, in order to obtain an optimal schedule, we need to focus only on transmission schedules in which all unsuccessful frames have at least a successful descendant.

We first make the following observation:

**Theorem 3.1.** Every schedule can be transformed into a schedule with only one-by-one frame trans-
mission, with at least the same successful frames and therefore, with at least the same reward function.

Proof. See Appendix 10.1.

Hence, it suffices to consider only one-by-one frame transmission in the search for an optimal schedule. In this case, the number of timeslots required to transmit frame $l$ is given by $\Delta t_l = \frac{2^S}{C}$. We assume the timeslot size is small enough such that $\Delta t_l$ are well approximated by integers.

Clearly, in terms of the reward function, there is no benefit in leaving a gap between the transmission of two consecutive frames. Therefore, in computing an optimal schedule, it suffices to consider only the transmission sequence, i.e., the set of frames selected for transmission and their order of transmission. With an optimal transmission sequence, an optimal sequence can be determined by simply transmitting the frames back-to-back without any gap between them. Hence, throughout the rest of this paper, a “transmission schedule” refers to only its “transmission sequence” without the timing information. Clearly, the subsequence relationship among transmission sequences is well defined.

The action space of this problem includes all permutations of all subsets of frames in the video sequence. Therefore, the complexity of exhaustive search would be prohibitive. In the next three chapters, we first present some inherent properties of an optimal transmission schedule, which will then be used in Chapters 5 and 6 to develop polynomial solutions to the scheduling problem.
Chapter 4

Properties of Optimal Schedule

In this chapter, we first describe a transformation of the dependency DAG of a video into rooted trees that indicate decodability. Based on this representation, we then present two general classes of HPS that are of interest, and give some important properties of optimal scheduling common to both classes and essential to the solutions presented in Chapters 5 and 6.

4.1 Modified Breadth First Search (MBFS) Trees

We adopt a version of the Breadth First Search (BFS) algorithm [34] on the video DAG $G$, which we call Modified Breadth First Search (MBFS). It takes $G$ and a node $s$ as input and outputs an MBFS tree rooted at $s$. The main difference with BFS is the following: at each node, instead of picking all unvisited children of that node, only decodable unvisited children are picked, where a node is decodable if and only if all of its ancestors have been visited. Moreover, in constructing the MBFS tree, the decodable unvisited children are sorted in ascending order of their deadlines. Such ordering is important to the concept of isomorphically ordered trees presented later. The MBFS algorithm is formally given in Algorithm 1.

We run MBFS on $G$ and each I-frame in the display order, creating an MBFS forest, whose components are MBFS trees rooted at the I-frames, each tree corresponding to a GOP. Let the maximum GOP size in the video sequence be $\Phi$. Then, the complexity of this procedure is $O(N\Phi \log \Phi)$, since the execution of line 13 of Algorithm 1 can be done with $\Phi \log \Phi$. Figure 4.1 illustrates the result of MBFS on the DAG in Figure 3.1 with node 16 removed. The main benefit of MBFS is to represent the dependency structure of frames in the format of trees.

We emphasize that in terms of the MBFS forest, the children of each node, in standard graph theoretic terminology for trees, is a subset of the children of that node in terms of the DAG $G$, as defined in Chapter 3. This is because being a child in the MBFS forest carries the additional requirement of being decodable. Furthermore, the set of descendants of an I-frame $I_i$ in terms of the DAG can be partitioned into two subsets in $GOP_{i-1}$ and $GOP_i$, where only the latter subset is the descendants of $I_i$ in terms of the MBFS forest. In the rest of this paper, when we refer to the parent-child and ancestor-descendant relations between frames, they are in terms of the DAG by default. When the relations are in terms of the MBFS forest, such exceptions will be clearly stated unless they are obvious from the context.

However, note that the set of descendants of a non-I-frame remains the same in terms of either the
Algorithm 1 Modified Breadth first Search(MBFS)(G,s)

1: for each node $u \in V[G] - \{s\}$ do
2: $\text{color}[u] \leftarrow \text{WHITE}$
3: $d[u] \leftarrow \infty$
4: $\pi[u] \leftarrow \text{NIL}$
5: end for
6: $\text{colors}[s] \leftarrow \text{GRAY}$
7: $d[s] \leftarrow 0$
8: $Q \leftarrow \emptyset$
9: ENQUEUE($Q, s$)
10: while $Q \neq \emptyset$ do
11: $u \leftarrow \text{DEQUEUE}(Q)$
12: $\text{Adj}[u] \leftarrow$ sequence of children of node $u$
13: sort $\text{Adj}[u]$ in ascending order of deadlines.
14: for each $v \in \text{Adj}[u]$ (in ascending order) do
15: if $(\text{color}[v] = \text{WHITE})$ and $(\forall i, i \to v \Rightarrow \text{color}[i] = \text{GRAY})$ then
16: $\text{color}[v] \leftarrow \text{GRAY}$
17: $d[v] \leftarrow d[u] + 1$
18: $\pi[v] \leftarrow u$
19: ENQUEUE($Q, v$)
20: end if
21: end for
22: $\text{color}[u] \leftarrow \text{BLACK}$
23: end while
24: return

MBFS forest or the DAG. Hence, if we focus our attention within a single GOP, then any frame has the same set of descendants in terms of either the MBFS forest or the DAG. In that case we only need to distinguish the reference terms of children but not those of descendants.

4.2 SIO and Quasi-SIO Classes

Based on the MBFS forest, we next give two important definitions that characterize the HPSs of interest.

Definition 4.1. The DAG representing the HPS of a video is called sequential if the ancestors of each node lie only on the path between the node and the root of the MBFS tree that contains the node.

Thus, if a DAG is sequential, then decoding a node only requires the availability of nodes residing on the path between the node and the root of the MBFS tree that contains the node. Clearly, if a video sequence consists of only I-frames and P-frames, then its DAG is sequential, since each node only has a single parent. The situation is more complicated when there are B-frames. From Figure 3.1, it is easy to see that the DAG in Figure 3.1 is sequential if and only if edges 16 $\rightarrow$ 14 and 16 $\rightarrow$ 15 are removed.

In the MBFS forest, let $T(x)$ denote the subtree rooted at node $x$, and $\min(T(x))$ and $\max(T(x))$ be the minimum and maximum display deadlines among the nodes in $T(x)$, respectively. Let $c_i(x)$ be the $i$th child of $x$.

---

1The following statement is a direct result of Definition 4.1: For an HPS video sequence sorted in the display order, consider two P-frames within a GOP. If they have a common descendant B-frame and the related DAG is sequential, then the preceding P-frame in the display order is the ancestor of the other P-frame. This is because otherwise the two P-frames would be irrelevant, so that they could not both reside on the same path to the common descendant B-frame from the root of its MBFS tree.
Chapter 4. Properties of Optimal Schedule

12

Definition 4.2. An isomorphically ordered tree is a rooted tree such that, if node \( x \) has \( k \) children, they can be re-ordered (and re-indexed) such that

\[
\max(T(c_i(x))) \leq \min(T(c_{i+1}(x))), \; 1 \leq i \leq k - 1 .
\] (4.1)

The above definition resembles that of B-trees [34], but there is no requirement for the tree being balanced. An example of isomorphically ordered trees are binary search trees. A more general example is shown in Figure 4.1. We confine our discussions to HPSs whose MBFS forest contain only isomorphically ordered trees. The following lemma will be used in deriving the optimal schedules.

Lemma 4.1. In an MBFS forest with isomorphically ordered trees, consider two irrelevant nodes \( x \) and \( y \). If \( d_x < d_y \), then \( \max(T(x)) < \min(T(y)) \).

Proof. See Appendix 10.2.

From the two definitions above, we say that an HPS is Sequential and Isomorphically Ordered (SIO) if its DAG is sequential and its MBFS forest contains only isomorphically ordered trees. For example, the HPS shown in Figure 3.1 is SIO if edges 16 → 14 and 16 → 15 are removed. Furthermore, to extend the concept of SIO to more general HPSs that contain B-frames that depend on an I-frame from the next GOP, we say that an HPS is Quasi Sequential and Isomorphically Ordered (quasi-SIO) if it would become SIO after such dependency was removed. For example, the HPS shown in Figure 3.1 is quasi-SIO.

In this work, we focus only on the SIO and quasi-SIO classes of HPSs, since many common HPSs in practice belong to either class. For example, the zero-delay structure [4] is SIO and the classical prediction structure [33] used as default in H.264/AVC is quasi-SIO. For a proof that the widely adopted hierarchical dyadic structure is quasi-SIO, see Appendix 10.3.

4.3 Canonical Form of Optimal Transmission Sequence

Theorem 4.1 below suggests that it suffices to find an optimal transmission sequence that satisfies some special properties. In particular, for the SIO and quasi-SIO classes, we say that a transmission sequence
that preserves the two properties in Theorem 4.1 is in the canonical form.

**Theorem 4.1.** In the SIO and quasi-SIO classes, any transmission sequence $S$ can be re-ordered into a transmission sequence $S'$ with at least the same set of successful frames as $S$; therefore, $z(S) \leq z(S')$, and with the following properties:

1. The ancestors of a frame are scheduled prior to that frame.
2. Consider two irrelevant frames $l_i$ and $l_j$ with $d_{l_i} < d_{l_j}$. If they are neighbors in the transmission schedule, then $l_i$ is scheduled before $l_j$. If $l_i$ is scheduled after $l_j$, then not all of the frames scheduled between $l_j$ and $l_i$ are irrelevant with respect to $l_j$.

**Proof.** See Appendix 10.4.

Theorem 4.1 will be used extensively in the rest of this paper. Furthermore, the lemmas below will also have important usage in Chapters 5 and 6.

**Lemma 4.2.** Under the properties in Theorem 4.1, consider the frames $f_1, o_1, \ldots, o_k$ such that $f_1 \in GOP_i$, none of $o_1, \ldots, o_k$ belongs to $GOP_i$, and they are all irrelevant with respect to $f_1$.

1. If they are scheduled in the order of $f_1, o_1, \ldots, o_k$, then

$$d_{f_1} < d_{o_r}, \quad 1 \leq r \leq k$$

and all frames $\{o_r\}_{r=1}^k$ belong to GOPs with indices greater than $i$.

2. If they are scheduled in the order of $o_k, \ldots, o_1, f_1$, then

$$d_{f_1} > d_{o_r}, \quad 1 \leq r \leq k$$

and all frames $\{o_r\}_{r=1}^k$ belong to GOPs with indices less than $i$.

**Proof.** See Appendix 10.5.

**Lemma 4.3.** Under the properties in Theorem 4.1, consider some frames $f_1, o_1, \ldots, o_k$ in an MBFS tree, which corresponds to a GOP. Suppose all of $o_1, \ldots, o_k$ are irrelevant with respect to $f_1$.

1. If they are scheduled in the order of $f_1, o_1, \ldots, o_k$, then

$$d_{f_1} < d_{o_r}, \quad 1 \leq r \leq k$$

2. If they are scheduled in the order of $o_k, \ldots, o_1, f_1$, then

$$d_{f_1} > d_{o_r}, \quad 1 \leq r \leq k$$

**Proof.** See Appendix 10.6.
Chapter 5

Optimal Schedule for the SIO Class

This chapter provides an algorithm to solve the Scheduling Problem for the SIO class. We first show that each transmission sequence has a unique canonical-form transmission sequence, which is a subsequence of some universal sequence. We then select an optimal schedule from the universal sequence through dynamic programming.

5.1 SIO Universal Sequence

First, we discuss how to find a canonical-form transmission sequence W given an arbitrary transmission sequence. The lemma below indicates that the frames of each GOP must be transmitted together in W. Moreover, due to the second property of Theorem 4.1, GOPs with earlier deadlines must be transmitted first.

Lemma 5.1. In an SIO canonical-form transmission sequence, for any i, no frame other than the frames of GOP_i can be scheduled between the frames of GOP_i.

Proof. Assume towards a contradiction that the lemma does not hold for some i. Then it is easy to see that there must exist two frames, f_1 and f_2, in GOP_i such that none of the frames scheduled between them belong to GOP_i. Let the frames between f_1 and f_2 be o_1, . . . , o_k. In the SIO class, all of these frames are irrelevant with respect to GOP_i. In particular, they are irrelevant with respect to f_1. As a result, Lemma 4.2 suggests o_k belongs to a GOP_j such that j > i, which implies that d_{f_2} < d_{o_k}. Noting that o_k and f_2 are neighbors, this violates the second property of Theorem 4.1.

For the frames inside a GOP, the properties of isomorphically ordered MBFS trees in Chapter 4 can be used to determine the order of transmissions. Lemma 5.2 indicates that all subtrees in an MBFS tree must be scheduled back-to-back in order to preserve Theorem 4.1.

Lemma 5.2. In an SIO canonical-form schedule, any frame scheduled between any two frames that are both in a subtree T must itself be in T.

Proof. See Appendix 10.7.

In W, consider an arbitrary node x and the subtrees T(c_1(x)), . . . , T(c_n(x)), where c_1(x), . . . , c_n(x) are the children of x in the MBFS forest sorted in ascending order of display deadlines. The frame x is
sent first according to the first property of Theorem 4.1. Then, by Lemma 5.2, the second property of Theorem 4.1, and isomorphic order, the frames in each $T(c_i(x))$ are sent together in the order of $c_i(x)$. Therefore, to determine the order of transmissions for each GOP in $W$, we should perform a generalized pre-order tree walk [34] on each tree of the MBFS forest, where the children of each node is visited in ascending order of display deadlines. This procedure uniquely determines $W$ and has a complexity of $O(N)$. Therefore, a canonical form transmission sequence is uniquely determined only by the set of selected frames for transmission. As an example, if the generalized pre-order tree walk is executed on the tree in Figure 4.1, the resulting sequence will be $0, 4, 2, 1, 3, 8, 6, 5, 7, 12, 10, 9, 11, 14, 13, 15$.

Next, consider the sequence of all frames of the original video. We call the canonical-form transmission sequence of this sequence the SIO universal sequence. Note that since we may hypothetically increase the link capacity until the entire video is successfully schedulable, such a transmission sequence always exist.

**Theorem 5.1.** For the SIO class of HPSs, the canonical-form of any transmission sequence is a subsequence of the SIO universal sequence.

**Proof.** Let $W$ be the canonical-form of a transmission sequence, with associated graph $G_W \subset G$. Since $G_W$ can be obtained from $G$ by removing the nodes that are not scheduled in $W$ and their descendants in the MBFS forest, the roots of trees in $G_W$ are roots in $G$ as well. Moreover, Lemma 5.2 indicates that a node and its descendants in the MBFS forest are scheduled together. This, together with the fact that the pre-order tree walk is executed on both $G$ and $G_W$, implies that $W$ can be obtained from the SIO universal sequence by removing subtrees, which corresponds to dropping frames.

Theorem 5.1 suggests that, given a link capacity limit, a transmission sequence that is a subsequence of the SIO universal sequence and which maximizes the playback quality is an optimal transmission sequence. This will be used in the next section to compute an optimal schedule.

### 5.2 Computation of Optimal Schedule

The following dynamic programming approach solves the Scheduling Problem for the SIO class. First, generate the SIO universal sequence and index its frames as $1, 2, \ldots, N$.

Then, define function $h(j, t)$ as the maximum reward function, if frames $\{j, \ldots, N\}$ are to be scheduled in the time interval $[t, \hat{T}]$ assuming all their parents with indices in the range from 1 to $j - 1$ (if any) are available. From the system model in Chapter 3, we have

$$z^* = h(1, 0)$$  \hspace{1cm} (5.1)

and the boundary conditions

$$h(N + 1, t) = 0, \quad \forall t \in \mathbb{Z} \hspace{1cm} (5.2)$$

$$h(j, t) = 0, \quad t > \hat{T}. \hspace{1cm} (5.3)$$

Furthermore, $h$ adheres to the following recursive equations, corresponding to the possible actions at
Chapter 5. Optimal Schedule for the SIO Class

16

time \( t \): 
\[
\begin{aligned}
    h(j, t) &= \max \left\{ 
        h(j+1, t+1), 
        q_j + h(j+1, t+\Delta t_j), 
        \begin{array}{ll}
            d_j - t & \geq \Delta t_j \\
            h(j+1, t+\Delta t_j), & d_j - t < \Delta t_j \\
            h(\min\{k : k > j, k \nsim \ j\}, t). & 
        \end{array}
    \right\}
\end{aligned}
\]  
\( (5.4) \)

In the above, the first term corresponds to the case where the optimal schedule for \( h(j, t+1) \) is also optimal for \( h(j, t) \), so no action is needed at \( t \). The second term corresponds to starting to transmit frame \( j \) at time \( t \), and the condition \( d_j - t \geq \Delta t_j \) ensures that the transmission of \( j \) will be successful. Similarly, the third term refers to starting to transmit frame \( j \) at time \( t \) but the condition \( d_j - t < \Delta t_j \) indicates it is not successful, although its transmission can help its descendants to achieve successful transmission. The forth term corresponds to dropping frame \( j \) and moving to inspect the next frame that does not depend on \( j \).

Since \( h \) only needs to be computed for \( 1 \leq j \leq N, 0 \leq t \leq \hat{T} \), dynamic programming requires \( O(N\hat{T}) \) processing time and \( O(N\hat{T}) \) memory to determine \( h(1, 0) \) and extract the optimal policy. Furthermore, since \( \hat{T} \) is linear in \( N \) in all practical video codecs with a constant frame rate, we have an overall complexity of \( O(N(\hat{T} + \Phi \log \Phi)) = O(N^2 + N\Phi \log \Phi)) \).

Now it will be shown it is possible to reduce the memory complexity with the cost of increasing the processing time complexity. In equation \( (5.4) \), the frame \( k^*_j \), which \( k^*_j = \min\{k : k > j, k \nsim \ j\} \), either is in the same GOP as the frame \( j \) or is the first DU of the next GOP; therefore, \( |k^*_j - j| \leq \Phi \). This point indicates if the goal is to figure out the values of \( h(0, 1) \) but not the optimal policy when calculating \( h(j, t) \), it is sufficient to have access to the values of \( h(k, t) \) for \( k = j + 1, \ldots, j + \Phi \). Therefore \( h(0, 1) \) can be figured out with the same processing time complexity but \( O(\Phi \hat{T}) \) memory. Furthermore, when \( h(0, 1) \) is reached, using the \( O(\Phi \hat{T}) \) stored data it is possible to restore part of opimal policy. Having this idea in mind and given the DUs sorted in canonical form, by dividing the DUs into blocks of size \( \Phi \hat{T} \) and executing the Dynamic programming for each block, it is possible to find the optimal policy with \( O\left(\frac{N^2\hat{T}}{\Phi}\right) \) processing time and \( O(\Phi \hat{T}) \) memory.
Chapter 6

Optimal Schedule for the Quasi-SIO Class

In this chapter, an optimal transmission sequence for the more general quasi-SIO class is presented, as an extension of the solution in Chapter 5.

6.1 Quasi-SIO Universal Sequence

Again, we will first show that all transmission sequences have unique canonical-form transmission sequences, and then they will be shown to be subsequences of a quasi-SIO universal sequence.

Let $V$ be a canonical-form transmission sequence of an arbitrary transmission sequence in the quasi-SIO class. From the first property of Theorem 4.1, $I_i$ will be scheduled prior to the frames in $N_i$ since $I_i$ is the ancestor of all of them, and similarly $I_{i+1}$ is scheduled prior to the frames in $D_i$.

Consider the following lemmas:

**Lemma 6.1.** In $V$, if $f_1$ and $f_2$ are two frames in $N_i$ such that no other frame in $N_i \cup \{I_{i+1}\}$ is scheduled between them, then no frame can be scheduled between them.

*Proof.* The proof is similar to that of Lemma 5.1. \hfill \qed

**Lemma 6.2.** In $V$, if $I_{i+1}$ is scheduled after $f_1 \in N_i$ such that no other frame in $N_i$ is scheduled between them, then no frame can be scheduled between them.

*Proof.* Towards a contradiction, suppose some frames $o_1, \ldots, o_k$ are scheduled between them. Since none of them are in $N_i$, they are all irrelevant with respect to $f_1$, so Lemma 4.2 indicates $d_{f_1} < d_{o_k}$ and $o_k$ belongs to some $GOP_j$ such that $j > i$. However, if $j = i + 1$, this will violate the first property of Theorem 4.1 since $o_k$ has appeared prior to $I_{i+1}$ in the schedule; if $j > i + 1$, then $d_{I_{i+1}} < d_{o_k}$, which violates the second property of Theorem 4.1. \hfill \qed

**Lemma 6.3.** In $V$, if $I_{i+1}$ is scheduled before $f_2 \in N_i$ such that no other frame in $N_i$ is scheduled between them, then no frame can be scheduled between them, and $f_2$ must be a descendant of $I_{i+1}$.

*Proof.* Towards a contradiction, suppose some frames $v_1, \ldots, v_k$ are scheduled between them. Since none of them are in $N_i$, they are all irrelevant with respect to $f_2$, so Lemma 4.2 indicates $d_{v_1} < d_{f_2}$ and $v_1$
Lemma 6.4. In $\mathcal{V}$, if $I_{i+1}$ is scheduled before $f \in \mathcal{N}_{i+1}$ such that no other frame in $\mathcal{N}_{i+1}$ is scheduled between them, then only frames in $\text{GOP}_i$ and frame $I_{i+2}$ can be scheduled between them.

Proof. Let the frames between $I_{i+1}$ and $f$ be $v_1, \ldots, v_k$. Two cases are considered: first, $I_{i+2}$ is among them, and second, it is not.

In the first case, Lemma 6.3 indicates $v_k = I_{i+2}$. Then, none of the frames $v_1, \ldots, v_{k-1}$ is an ancestor of $v_k$. Furthermore, the first property of Theorem 4.1 suggests that none of them can be a descendant of $v_k$ either. Hence, all frames $v_1, \ldots, v_{k-1}$ are irrelevant with respect to $v_k$, and Lemma 4.2 indicates they all belong to GOPs with indices less than $i + 2$. Since $v_1, \ldots, v_k$ do not belong to $\mathcal{N}_{i+1}$, they all belong to GOPs with indices less than $i + 1$. The same result can be obtained in the second case, with Lemma 4.2 applied to frame $f$ instead of $v_k$.

Furthermore, in both cases, $v_1$ cannot be irrelevant with respect to $I_{i+1}$, since that will violate the second property of Theorem 4.1. Then the first property of Theorem 4.1 indicates $v_1$ must be a descendant of $I_{i+1}$. As a result, $v_1 \in \mathcal{N}_i$ in both cases. Now let $\xi = \max\{i : 1 \leq i \leq k, v_i \in \mathcal{N}_i\}$. Lemma 6.1 indicates that $v_1, \ldots, v_\xi$ all belong to $\mathcal{N}_i$.

In the first case, suppose $\xi < k - 1$. Then the frames $v_\xi, \ldots, v_{k-1}$ belong to neither $\text{GOP}_i$ nor $\text{GOP}_{i+1}$. Therefore, they are all irrelevant with respect to $v_\xi$, and due to Lemma 4.2, they all belong to GOPs with indices higher than $i$. In addition, since none of them belong to $\mathcal{N}_{i+1}$, they all belong to GOPs with indices higher than $i + 1$. But this contradicts the result above, so $\xi = k - 1$. Similarly, in the second case, we can show that $\xi = k$.

Lemmas 6.1, 6.2, and 6.3 jointly indicate that the frames of $\mathcal{M}_i = \mathcal{N}_i \cup \{I_{i+1}\}$ must be sent together. Moreover, Lemma 6.4 indicates that $\mathcal{M}_{i+1}$ must be scheduled immediately after $\mathcal{M}_i$, for all $i$.

Next, we determine the transmission order of frames inside each $\mathcal{M}_i$. Due to the first property of Theorem 4.1, each I-frame $I_{i+1}$ is scheduled prior to $\mathcal{D}_i$ in $\mathcal{V}$. With respect to $I_{i+1}$, the frames of $\text{GOP}_i$ can be divided into two groups: $\mathcal{A}_i$, the frames scheduled prior to $I_{i+1}$, and $\mathcal{B}_i$, the frames scheduled after $I_{i+1}$. Clearly all frames of $\mathcal{D}_i$ are in $\mathcal{B}_i$. Since the frames in $\mathcal{M}_i$ must be sent together, the frames in $\mathcal{A}_i$ are sent together, and the same holds for $\mathcal{B}_i$. The definition of quasi-SIO indicates that the frames of $\mathcal{A}_i$ can be scheduled by Lemma 5.2 using the subgraph of $G$ which comprises the nodes of $\mathcal{A}_i$. Moreover, for $\mathcal{B}_i$ since $I_{i+1}$ has already been scheduled, the frames of $\mathcal{B}_i$ can be scheduled similarly based on the isomorphic ordering property, the second property of Theorem 4.1, and Lemma 5.2. Hence, all subtrees in $\mathcal{B}_i$ are transmitted as contiguous blocks in the ascending order of the display deadlines of their roots.

It remains to determine the relative position of the frames in $\mathcal{N}_i - \mathcal{D}_i$ with respect to $I_{i+1}$. Let a critical node be defined as $f$, $f \in \mathcal{D}_i$, such that in the path between $f$ and the root of the MBFS tree containing $f$, no other frame in $\mathcal{D}_i$ appears. An example of this is depicted in Figure 6.1, where $l_i^1, l_i^2$ and $l_i^3$ are critical nodes. It is easy to see that a critical node in $\text{GOP}_i$ is a child of $I_{i+1}$, but a child of $I_{i+1}$ in $\text{GOP}_i$ is not necessarily a critical node. Let $\Gamma_i$ be the set of all critical nodes. It is clear from Figure 6.1 that the set of nodes in the subtrees rooted at the critical nodes of $\text{GOP}_i$ is equivalent to $\mathcal{D}_i$.

Furthermore, due to the definition, the members of $\Gamma_i$ are pair-wise irrelevant. Let $\zeta_i$ be the member of $\Gamma_i$ with the smallest display deadline.
Lemma 6.5. In $\mathcal{V}$, frame $\zeta_i$ must be scheduled immediately after $I_{i+1}$.

Proof. See Appendix 10.8.

Furthermore, we have the following result.

Lemma 6.6. In $\mathcal{V}$, consider frame $f \in \mathcal{N}_i - \mathcal{D}_i$ that is irrelevant with respect to $\zeta_i$.

- If $d_f < d_{\zeta_i}$, $f$ is scheduled before $I_{i+1}$.
- If $d_f > d_{\zeta_i}$, $f$ is scheduled after $I_{i+1}$.

Proof. See Appendix 10.9.

The above observations indicate that for any quasi-SIO transmission sequence, if we first ignore the backward prediction edges from $I_{i+1}$ to B-frames in $\text{GOP}_i$, for all $i$, and create the SIO canonical form transmission sequence, then the quasi-SIO canonical form schedule can be obtained by simply moving $I_{i+1}$, for all $i$, in the SIO canonical form transmission sequence backward so that it is positioned immediately prior to the earliest frame of $\mathcal{D}_i$ in the transmission order. Furthermore, $\mathcal{V}$ is uniquely determined given any transmission sequence. As an example, the quasi-SIO canonical form sequence referring to the quasi-SIO prediction structure in Figure 3.1 is $0, 4, 2, 1, 3, 8, 6, 5, 7, 12, 10, 9, 11, 16, 14, 13, 15$. Hence, we define the quasi-SIO universal sequence similarly to the SIO universal sequence in the previous chapter.

Then, similar to Section 5, we have the following theorem, which will be exploited to derive the optimal schedule given a link capacity limit.

Theorem 6.1. For the quasi-SIO class of HPSs, the canonical form of any transmission sequence is a subsequence of the quasi-SIO universal sequence.

Proof. The proof is similar to that of Theorem 5.1. We only need to additionally consider the case where an I-frame $I_{i+1}$ is not included in the transmission schedule. We note that the removal of $I_{i+1}$ induces the removal of entire blocks $\mathcal{D}_i$ and $\mathcal{N}_{i+1}$, which does not alter the canonical form.
6.2 Computation of Optimal Schedule

First, generate the quasi-SIO universal sequence and index the frames by 1, 2, ..., N.

Then, define function \( g(j, t, s) \), similarly to \( h(j, t) \), as the maximum sum quality of successful frames, if frames \{j, ..., N\} are to be scheduled in the time interval \([t, \hat{T}]\) assuming all their parents with indices in the range from 1 to \( j - 1 \) (if any) are available. The additional parameter \( s \) is an integer in \{0, 1, 2, 3\}, whose binary representation specifies the status of the two nearest I-frames in the quasi-SIO universal sequence that precede \( j \), i.e., \( s = (x_1(j), x_2(j)) \), where \( x_2(j) = 1 \) if the nearest preceding I-frame is selected for transmission and 0 otherwise, and \( x_1(j) \) is similarly defined concerning the second nearest preceding I-frame.

Similarly to Section 5.2, we have

\[
  z^* = g(1, 0, 0),
\]

with boundary conditions

\[
  g(N + 1, t, s) = 0, \quad \forall t \in \mathbb{Z}, \forall s \quad (6.2)
\]

\[
  g(j, t, s) = 0, \quad t > \hat{T}. \quad (6.3)
\]

The recursive equations for \( g(j, t, s) \) are given below, with special consideration for the parameter \( s \).

1) If \( j \) is not an I-frame, let the index of the GOP that \( j \) belongs to be \( i_j \). Three cases are considered:

1a) Suppose \( j \in A_{i_j} \). This is the case where \( j \) does not depend on \( I_{i_j+1} \) and is located prior to \( I_{i_j+1} \) in the quasi-SIO universal sequence. If \( I_{i_j} \), the nearest preceding I-frame, is transmitted, then the outcome is similar to Section 5.2; otherwise, \( j \) should not be selected. Hence, for \( s = 1, 3 \), we have

\[
  g(j, t, s) = \max \left\{ g(j, t + 1, s), q_j + g(j + 1, t + \Delta t_j, s), g(j + 1, t + \Delta t_j, s), g(\min\{k : k > j, k \not\approx j\}, t, s) \right\} \quad (6.4)
\]

and for \( s = 0, 2 \), we have

\[
  g(j, t, s) = \max \left\{ g(j, t + 1, s), g(\min\{k : k > j, k \not\approx j\}, t, s) \right\} \quad (6.5)
\]

1b) Suppose \( j \in D_{i_j} \). This is the case where \( j \) depends on both \( I_{i_j} \) and \( I_{i_j+1} \) and is located behind both in the quasi-SIO universal sequence. If both I-frames are transmitted, then the outcome is similar to Section 5.2; otherwise, \( j \) should not be selected. Hence, for \( s = 3 \), we use (6.4), and for \( s = 0, 1, 2 \), we use (6.5).

1c) Suppose \( j \in B_{i_j} - D_{i_j} \). This is the case where \( j \) does not depend on \( I_{i_j+1} \) but is located behind \( I_{i_j+1} \) in the quasi-SIO universal sequence. If \( I_{i_j} \), the second nearest preceding I-frame, is transmitted, then the outcome is similar to Section 5.2; otherwise, \( j \) should not be selected. Hence, for \( s = 2, 3 \), we use (6.4), and for \( s = 0, 1 \), we use (6.5).

2) If \( j \) is an I-frame, then we also need to update \( s \) besides scheduling the frame. Hence,
where $\tilde{I}_j$ is the next I-frame after $j$, and we update the two bits of $s$ depending on the scheduling outcome.

In the above dynamic programming formulation, state $s$ needs to take only four values due to the fact that the frames of a GOP only depend on at most two I-frames. Therefore, we again have complexity $O(N(\hat{T} + \Phi \log \Phi)) = O(N^2 + N\Phi \log \Phi)$. The memory complexity can be reduced similar to Section 5.2.
Chapter 7

Multi-layer Transmission Extension

In this chapter a suboptimal scheme for the transmission of multi spatial layer video is proposed. First, the optimal scheme presented in the previous chapters will be extended to videos where frames can have the same deadline. Second, by presenting the concept of virtual deadlines, the suboptimal scheme is presented.

7.1 Simultaneous Video Transmission

In this section, the optimal solutions proposed in the previous chapters will be extended to cover video sequences with the possibility of the same frame display deadlines. In particular, in this section, we make the following assumption about video sequences. In the display sequence, if $i < j$, then $d_i \leq d_j$.

This is in contrast to the previous chapters where we assumed frames have distinct deadlines. Now in line 13 of Algorithm 1, for frames with the same deadlines, sort in the ascending order of display indices.

For the generalized scheduling problem (no relation between the number of frames and the deadlines), there is the following theorem:

**Theorem 7.1.** The Generalized Scheduling Problem is NP-complete.

**Proof.** Please see Appendix 10.10.

7.1.1 The Cousin Relationship

Consider again the the lemma used in the proof of Lemma 4.1 which is restated as follows:

**Lemma 7.1.** Let $x$ and $y$ be two irrelevant nodes in an isomorphically ordered tree. A unique node $w$ and unique indices $i$ and $j$, $i \neq j$, exist such that $x$ belongs to $T(c_i(w))$ and $y$ belongs to $T(c_j(w))$.

**Proof.** See Appendix 10.11.

Based on lemma 7.1, we can define the cousin relationship as follows.

**Definition 7.1.** Let $x$ and $y$ be two irrelevant nodes in an MBFS Forest and let the GOPs that $x$ and $y$ belong to be $n_x$ and $n_y$ respectively. We say $xCy$ and $x$ is called the left cousin of $y$ and $y$ is called the right cousin of $x$ if and only if one of these two conditions is satisfied:
• $n_x < n_y$.

• If $n_x = n_y$, there is a specific node $w$ and indices $i$ and $j$ such that $x$ belongs to $T(c_i(w))$ and $y$ belongs to $T(c_j(w))$ and $i < j$.

For the condition $n_x = n_y$, Lemma 7.1 ensures the frame $w$ and indices $i$ and $j$ always exist. An immediate result of the above definition is

$$xCy \Rightarrow d_x \leq d_y$$

(7.1)

It is easy to verify the left (right) cousin relationship is not symmetric. Furthermore, the cousin relationship has the following properties:

**Lemma 7.2.** Within an MBFS tree, the cousin relationship, $C$, is transitive.

**Proof.** See Appendix 10.12.

**Lemma 7.3.** Suppose three frames $l_1$, $l_2$ and $l_3$ have the following features:

• $l_1Cl_2$

• $l_2Cl_3$

• $l_1$ and $l_3$ are irrelevant.

Then, $l_1Cl_3$

**Proof.** See Appendix 10.13.

**Lemma 7.4.** In an MBFS forest with isomorphically ordered trees, consider two irrelevant nodes $x$ and $y$. If $xCy$, then for all nodes $l_1$ and $l_2$ in the set of nodes of $T(x)$ and $T(y)$ respectively:

$$l_1Cl_2$$

(7.2)

Hence, as a result: $\max(T(x)) \leq \min(T(y))$

**Proof.** See Appendix 10.14.

Now, according to the above definition and lemmas, if in Chapter 4, starting from Section 4.3, the following changes are made, it is easy to verify that all statements and proof still hold:

• In any comparison between two frame display deadlines, perform the following changes:
  1. change “<” to “C” and replace deadlines with the actual frames
  2. change “greater” to “right cousin”
  3. change “smaller” to “left cousin”
  4. change “smallest” to “having no left cousin”

• Instead of Lemma 4.1, use Lemma 7.4.

Therefore, the optimal solutions in the previous chapters is extended to video sequences which can have frames with the same deadline. However, note that the proof of the modified Theorem 4.1 needs additional explanation. So, for the matter of completeness, this proof is restated below:
Proof of the modified Theorem 4.1

Given a transmission sequence that does not necessarily meet the two properties, it will be modified to meet those properties in two steps.

First Step: With the assumption that the first property does not hold, there must be frames \( l_1 \) and \( l_2 \) such that frame \( l_1 \) is the ancestor of frame \( l_2 \) but \( l_2 \) is scheduled before \( l_1 \). An example of this is shown in Figure 10.6(a). Let \( K_{l_2} = \{ \text{descendants of } l_2 \text{ scheduled after } l_2 \text{ and before } l_1 \} \). Also let the index of the timeslot in which the transmission of \( l_2 \) starts be \( e_1 \) and the index of the timeslot in which the transmission of \( l_1 \) ends be \( e_2 \). Recall at least one member of the set \( K_{l_2} \) has to be successful; otherwise, the entire set can be dropped with no effect on the reward function. In addition, the transmission of frame \( l_1 \) has to terminate prior to the deadline of its successful descendants. As a result:

\[
e_2 \leq d_l, \forall l \in K_{l_2} \cup \{l_2\}, \text{ } l \text{ is successful} \tag{7.3}
\]

Now consider the frames in the interval \([e_1, e_2]\). They can be divided into three sets: \( \{l_1\}, A_1 = K_{l_2} \cup \{l_2\}, \text{ and } A_2 = \{ \text{All the frames in interval } [e_1, e_2] \} - (A_1 \cup \{l_1\}) \). Now starting from timeslot \( e_1 \) first schedule \( A_2 \), then \( l_1 \), and then \( A_1 \), such that the order of frames in \( A_2 \) in the original sequence is preserved and also the same for \( A_1 \). An example of this operation is shown in Figure 10.6(b).

Any previously successful frame in the time interval \([e_1, e_2]\) remains successful. In particular, since all frames in \( A_2 \) either move backward in time or do not change position, the previously successful ones remain successful. Moreover, due to inequality (10.1) all previously successful frames in \( A_1 \) remain successful as well. Furthermore, no new violation of the first property is created. Since this operation always repairs at least one pair that violates the first property, and the number violations is finite, repetition of the operation will end in an solution which satisfies the first property.

Second Step: Due to lemma 4.1, \( d_{l_i} \leq d_{l_j} \). Let \( U \) be the set of frames scheduled between \( l_j \) and \( l_i \). Suppose all of them are irrelevant with respect to \( l_j \). Let \( t_0 \) be the timeslot when the transmission of \( l_j \) starts and \( t_1 \) be the timeslot when the transmission of \( l_i \) ends. If \( l_i \) is successful, then \( t_1 \leq d_{l_i} \leq d_{l_j} \).

Else, if \( l_i \) is not successful, then it has to have a successful descendant, namely \( f_i \). Let \( t_2 \) be the timeslot when the transmission of \( f_i \) ends. Since \( f_i \) is successful: \( t_2 \leq d_{f_i} \). Furthermore, from the previous step, we have \( t_1 < t_2 \). We next consider two cases based on whether \( l_i \) is an I-frame. If \( l_i \) is not an I-frame, then \( f_i \) belongs to the same tree as \( l_i \) in the MBFS forest. Hence, Lemma 4.1 indicates \( d_{f_i} \leq d_{l_j} \). If \( l_i \) is an I-frame, let \( n_{l_i} \) be the index of the GOP that \( l_i \) belongs to. Then since \( l_i \) is an I-frame, the frame \( l_j \) has to belong to a GOP with an index higher than \( n_{l_i} \). But, the children of \( l_i \) are only in \( GOP_{n_{l_i} - 1} \) and \( GOP_{n_{l_i}} \). As a result, \( d_{f_i} \leq d_{l_j} \). Hence, in both cases, we have \( t_1 < t_2 \leq d_{l_j} \). According to this result, it should be mentioned it is not possible to have \( d_{l_i} = d_{l_j} \) and frame \( l_i \) be unsuccessful at the same time. The reason is in this case, the recent inequality indicates \( t_2 < d_{l_i} \); but if \( l_i \) is not successful, then \( d_{l_i} < t_1 < t_2 \leq d_{f_i} \) which is a contradiction.

Now, we can re-order the frames starting at \( t_0 \) as follows: \( U \), then \( l_i \), and then \( l_j \). This clearly will
not harm $l_i$ and $l_j$. In addition, the frames of $U$ are moved backward in time, so none of them is harmed either. Furthermore, none of the frames outside of $[t_0, t_1]$ are affected. This operation is illustrated in Figure 10.7. A special case of this is when $U$ is empty, i.e., when $l_j$ and $l_i$ are neighbors.

### 7.2 Analysis of Multi-layer SVC transmission

In this section, a suboptimal scheduling scheme for the transmission of multi spatial layer video is proposed. Since coarse grain scalability (CGS) in the SVC standard is a special case of spatial scalability, it is covered in this section as well. In a multiple spatial layer video, each frame has a related Data Unit (DU) in each spatial layer which corresponds to a specific resolution quality of that frame. The DUs have the same display deadline as the original related frames. The DUs in higher spatial layers adhere to the same HPS that the base layer preserves. In addition, each DU can depend on the DU with same display index in the preceding layer. Figure 7.3 shows an example of a multilayered structure with $G_{16B15}$.

Index the layers with $0, \ldots, L - 1$, where $L$ depicts the number of spatial layers and 0 corresponds to the base layer. By examining the DAG comprising of all layers, it is clear it preserves neither the sequential property nor the isomorphically ordered property. In particular, each B-frame DU in higher layers will depend on at least three pair-wise irrelevant DUs: First, the DU with the same display index in the previous layer; second, ancestor DUs in the same layer. It does not even belong to the quasi-SIO class. Therefore, the canonical form may no more be unique; furthermore, it may not even exist. So, if a dynamic programming approach is to be used, the number of the states to be addressed can increase hugely in both the number of frames of each GOP and the number of the layers.
So instead, a sub-optimal approach is proposed; priorities are assigned to the spatial layers for scheduling. In particular starting from the base layer, each spatial layer is considered at a time and is completely scheduled regardless of other layers. Then if any resources in terms of idle timeslots is left, they are used for the transmission of higher layers. In other words, the multi-layer scheduling scheme is executed layer by layer and starting form the lowest layer.

In the remainder of this section we will use the following terms and notations. Let \( \{dl_i\}_{i=0}^{N} (d_0 = 0) \) be the sequence of DU deadlines sorted in ascending order. Moreover let \( S \) be the transmission schedule. Initially, \( S \) is empty and each time a layer is scheduled, \( S \) is updated. When considering scheduling the \( k-th \) layer, each DU \( l \) in that layer is associated with \( (S_l, d^k_l, q_l) \) where the first and last term are the same as before and the second term is the virtual deadline which is assigned before starting scheduling the \( k-th \) layer. More details of the virtual deadlines will be presented later.

Now suppose the \( k-th \), \( (k \geq 0) \) layer is to be scheduled given layers 0, \ldots, \( k-1 \) are already scheduled (In the case of \( k = 0 \), no layer is previously scheduled). Scheduling of the \( k-th \) layer consists of two steps which is discussed as follows:

### 7.2.1 Step 1: Layer-wise Sequencing

Discard all DUs which their corresponding DUs in the previous layers have been dropped or they are unsuccessful (For the case of \( k = 0 \), no DUs are dropped). Next, given the MBFS forest of the remaining DUs in \( k-th \) layer , find the optimal transmission schedule using the single-layer solutions provided in section 5.2 or 6.2 based on the case. This transmission schedule will be called \( s_k \). The values of \( \{d^k_i\}_{i=1}^{N}, k > 0 \) are assigned at the end of the second step. However as initial values let: \( \forall 1 \leq i \leq N, d^0_i = dl_i \).

### 7.2.2 Step 2: Resource Extraction and Deadline Update

In this step, algorithm 2 is performed on the already scheduled sequence of the \( k-th \) layer.

---

**Algorithm 2** Resource Searcher(RS)(k)

1: \( s_k = \) schedule of layer \( k \)
2: \( Len=\) size\((s_k)\)
3: for \( i = Len \) to 1 do
4: if \( (B(i) + \Delta t_i \leq d_i) \) then
5: \( scs \leftarrow 1; \) scs shows whether frame \( i \) is successful
6: end if
7: while True do
8: \( Y = B(i) + \Delta t_i \)
9: if \( (Y == B(i+1)) \) then
10: Break
11: end if
12: if \( (scs == 1) \) and \( (Y == d_i) \) then; If frame \( i \) is successful, it has to remain successful
13: Break
14: end if
15: \( B(i) \leftarrow B(i) + 1 \)
16: end while
17: end for

\( B(\cdot) \) indicates the transmission start timeslot of a DU. The algorithm \( RS(k) \) extracts the available
resources for the schedule of the $k - th$ layer. In particular, it delays the transmission of selected DUs as long as they do not reduce the reward function. Unsuccessful DUs are moved until their transmission end time meets the transmission start time of the next DU. Delaying preceding layers using algorithm 2, while ensuring the successful transmission of previously successfully scheduled DUs, maximizes the opportunity of successful transmission for higher layers in the remaining idle timeslots.

Once $RS(k)$ is executed, define the sequence $\{\tau_k^i\}_{i=1}^N$ as follows. $\tau_k^i$ is the smallest transmission start time of the DUs partially or completely scheduled in the interval $[d_{l_i-1}, d_{l_i}]$. If there is no such DU, then let $\tau_k^i = d_{l_i}$.

Now, define the sequence $\{\Delta x_k^i\}_{i=1}^N$ as follows:

$$\Delta x_k^i = \max\{d_{l_i-1}, \tau_k^i\} - d_{l_i-1} \quad (7.4)$$

The term $\Delta x_k^i$ presents the length of the idle period in the interval $[d_{l_i-1}, d_{l_i}]$ given layers $0, \ldots, k-1$ are already scheduled. In this definition, the $\max$ term presents the end of the idle period and the start of the busy period in the interval $[d_{l_i-1}, d_{l_i}]$. The maximum is used to take account for the situation in which the interval $[d_{l_i-1}, d_{l_i}]$ is full. This idle period could be used as resource for the scheduling of the next layers. This concept is depicted in figure 7.4.

Now define the sequence $\{d_{k+1}^i\}_{i=1}^N$ as follows:

$$d_{k+1}^i = \sum_{j=1}^{i} \Delta x_k^j \quad (7.5)$$

The elements of this sequence are called virtual deadlines and they are used as deadlines for the first step of the next layer. Defined $T_k$ be the time horizon referring to the virtual deadlines of the $k - th$ layer. Furthermore, let $T$ refer to the time horizon of the actual deadlines. Therefore, $s_k$ refers to $T_k$ and $S$ refers to $T$. In fact, for all values of $k$, $T_k$ is a sub-set of $T$.

At the end of the second step, the transmission schedule $S$ is updated. In particular, each virtual deadline $d_k^i$ in $T_k$ corresponds to the timeslot $d_{l_i-1} + \Delta x_k^{i-1}$ in $T$. From another point of view, each
interval $[d_i^k, d_{i+1}^k]$ in $T_k$ corresponds to the idle period of $[d_{li}, d_{li+1}]$ in $T$ given layers $0, \ldots, k-1$ are already scheduled and included in $S$. So, given the above explanation, $S$ can be updated using $s_k$. It should be mentioned this update may cause some DUs be scheduled preemptively, since they may not cover tightly one or more of the $[d_i^k, d_{i+1}^k]$ intervals and the corresponding intervals in $T$ may not be neighbors. However, using the mechanism indicated in Appendix 10.1 and figure 10.1, a preemptive solution can be transformed to a non-preemptive solution.

The execution of the algorithm will terminate when all layers are scheduled. The complexity of the presented algorithm is $O(N(L\bar{T} + \Phi \log \Phi))$. 
Chapter 8

Experiment with Video Traces

8.1 Single Layer Transmission Simulation

Even though the proposed schedule has been proven optimal, we are interested in its numeric performance when applied to real video traces.

8.1.1 Methodology

The proposed scheme is simulated in Matlab, using H.264 video traces provided by [33]. We present here results for the “NBC News (352 × 288)” and “SONY (1920 × 1080)” traces, representing faster moving and slower moving scenes, respectively. We take the first 305 frames of each video. Both videos use the G16B15 hierarchical dyadic structure at 30 frames per second. The luminance of a frame is used as its quality measure, so the objective function averaged over the number of frames represents the average Y-PSNR at the receiver [9].

We compare the optimal solution with two suboptimal alternatives. The first is a best-effort earliest-deadline-first (EDF) algorithm, in which the frames are scheduled in their display order, and the frames that cannot be successful are dropped by the transmitter. EDF is the traditional scheme used for video streaming [6]. The second is a priority-based earliest-deadline-first (PBEDF) algorithm, which combines frame prioritization [13][14] with EDF. In PBEDF, the set of frames are partitioned into blocks of size $M$. In each block, the I-frames are scheduled first in EDF order, and the B-frames are scheduled last in EDF order. For fair comparison, in the results below we always use an optimal $M$ obtained by exhaustive search.

8.1.2 Experimental Results

Video playback quality is dictated by the channel rate and the initial playback delay, which is defined as the time the receiver waits before video playback while receiving data. In Figure 8.1, we consider different initial delays of 0.1, 1, and 5 seconds, and for each initial delay, we study all values of link capacity $C$ up to the point when all algorithms give lossless performance.

We observe that, for a wide range of parameter settings, the optimal schedule substantially outperforms the sub-optimal alternatives. The exception is only under extremely relaxed environments, e.g.,
Figure 8.1: Y-PSNR vs. link capacity $C$ for single spatial layer diagrams.
when the initial delay is large or when the link capacity is high. In practical video streaming, where the users are inpatient, and the network bandwidth limited, the benefit of the optimal schedule is apparent. In addition, under the optimal schedule, the Y-PSNR is monotonically increasing in the link capacity as expected. This is not the case for EDF and PBEDF, which results from the dependence structure. In particular, a slightly higher link capacity may drive EDF and PBEDF to over zealously transmit a frame, which in turn reduces the time left to transmit an ancestor that is later in the display order, possibly rendering the ancestor unsuccessful. Hence, with these sub-optimal schemes, increasing the link capacity is not always beneficial.

8.2 Multi-layer Transmission Simulation

8.2.1 Methodology

Similarly to the single layer transmission, the proposed scheme for multi layer transmission is simulated in MATLAB, using CGS SVC video traces provided by [33]. The results are presented for two video traces, namely “SONY_DEMO(352 × 288)” and “Tokyo_Olympics(352 × 288)”. The first 193 frames of each video is taken. Both videos use G16B15 hierarchical dyadic structure and have two enhancement layers at a frequency rate of 30 frames per second. In addition, delta quantization parameter (DQP) of the first and second video traces are 15dB and 10dB respectively.

The quality measures assigned to the base layer DUs are similar to the single layer transmission simulations. For the DUs of an enhancement layer, the difference between the luminance of the related frame in that spatial layer and the previous spatial layer is used as its quality measure.

The proposed scheme is compared with multi-layered version of the EDF and PBEDF schemes. In the multilayer version of EDF, all the DUs refering to a frame are sent together starting from the lowest to the highest layer and any DU which can not meet its deadline is discarded. In the multi-layer version of PBEDF, each spatial layer is partitioned and reordered separately and similarly to the single layer transmission PBEDF with the parameter $\bar{M}$. Then the corresponding blocks of different spatial layers are sent together starting from the lowest to the highest layer. Again the optimal $M$ is found through exhaustive search.

8.2.2 Experimental Results

The simulations are carried out for the initial delays of 0.5, 2 and 5 seconds as displayed in figure 8.2. Similar to the single layer case, the simulations are performed for all values of channel rate up to the value where all scheme achieve lossless performance.

According to the diagrams, at very low channel rates, the designed policy and the other alternatives differ substantially in performance. However, at higher channel rates, the performance of PBEDF becomes comparable to the designed policy. This fact is due to the small size of the base layer DUs compared to the single layer transmission case.
Figure 8.2: Y-PSNR vs. link capacity $C$ for multi spatial layer diagrams.
Chapter 9

Conclusions, Extensions and Future Work

9.1 Conclusions

In this thesis, efficient transmission schemes for video streaming were presented which attempt to exploit the dependency structure among the video data.

For the transmission of video with a single spatial layer, a frame scheduling scheme has been developed for optimal video streaming over a link with limited capacity. For two general classes of HPSs, it is shown that an optimal transmission sequence can be found as a subsequence of some optimal universal sequence in term of frame indices; the derivations in this work clearly indicate this is not a trivial result. Efficient dynamic programming solutions are proposed to identify the optimal schedules with polynomial complexity.

Furthermore, for the transmission of video with multiple spatial layers, the optimal schemes for transmission of video with a single spatial layer were extended to the scenario where video frames are allowed to have the same display deadlines. This result was achieved by defining and using the cousin relationship. Then, a sub-optimal scheme for the transmission of video with multiple spatial layers was proposed which uses the concept of virtual deadlines. Experiments with video traces show that the proposed schedules can substantially improve the playback quality over existing alternatives.

9.2 Extensions

Although the proposed algorithms in this thesis were developed in the context of video streaming, they have general utility beyond video streaming. In particular, they can be applied to any scheduling problem with deadlines, dependency and a reward function which is the sum of per-job rewards.

Furthermore, although in this work a channel with constant rate was addressed, the developed schemes apply to channels with non-uniform and non-causal rate. In particular, if the state of the channel during the an entire video streaming session is pre-known, it is possible to transform the problem to a problem with constant channel rate. More specifically, if instead of scaling the packets sizes with the channel capacity, the time horizon is scaled with the channel capacity, the time horizon can be transformed to the bit horizon and the deadlines can be adjusted respectively. As a result, the problem
is transformed to a constant channel rate scenario. From this point, a possible direction would be to extend the solution to channels with random capacity, i.e., not previously known and non-constant channels. This would help to address wireless channels.

9.3 Future Work

The developed models and the presented transmission schemes represent a new approach for understanding and designing efficient schemes to support video streaming applications. From this thesis, several potential directions can be suggested as follows:

- The transmission schemes presented in this thesis, targeted pre-stored video. A possible direction would be to study efficient transmission schemes for broadcasting and video conferencing. The difference is that in these applications, the frames (DUs) will be associated with release times as well as deadlines, that must be incorporated in design of the schemes. Due to the release times, the frame (DU) information is not initially known, so designing offline schemes is not possible for these scenarios.

- Investigating dynamic encoding and error concealment techniques jointly with transmission scheduling is another possible direction which would result in a more practical realization of a video streaming application.

- In this thesis, the constant rate channel model was assumed to be supported by lower layer protocols. A potential direction would be to design schemes addressing the constraints of the lower layer protocols, especially the physical layer. The extension of the work of this thesis to erroneous and random capacity channels will make it possible to address wireless video streaming.

- The focus in this thesis was on designing schemes for a single video streaming session. Another possible direction will be to address handling multiple video streaming sessions over a shared channel.
Chapter 10

Appendices

10.1 Proof of theorem 3.1

There are two possible scenarios where the frames are not transmitted one-by-one in some part of a schedule. First, the transmission of a frame may be interrupted by the transmission of parts of other frames (preemptive transmission). Second, parts of multiple frames are transmitted simultaneously. We present in the following how to transform the schedule in both scenarios to one-by-one frame transmission without incurring new delay in the transmission end time of any frame.

In the first scenario, as illustrated in Figure 10.1, supposed $A_1$ and $A_2$ are consecutive fragments of a frame, and some other frames (or parts of frames) are scheduled between them. Clearly, swapping the transmission of $A_2$ and that of the interrupting frames, as depicted in the right side of Figure 10.1, will not cause new delay in the transmission end time of any frame. Therefore, all preemptive transmissions can be removed in this way.

In the second scenario, we first consider the case where frames $l_1, \ldots, l_n'$ exactly overlap each other, within the transmission interval $[t_1, t_2]$. As depicted in Figure 10.2, if these frames are sent one-by-one under the same link capacity, then clearly the amount of data transmitted in this interval will not change and all the frames can be transmitted before time $t_2$. Furthermore, the successful frame parts in this interval remain successful for the least. Next, consider the case where the frames partially overlap each other. In this case, time can be divided into subintervals, such that in each of them, the frame segments completely overlap each other. Then the same transformation above can be carried out in each subinterval, to produce a schedule without simultaneous transmissions but with preemptive transmissions. This is then reduced to the first scenario.

![Figure 10.1: Transforming a preemptive schedule to a non-preemptive schedule.](image)
10.2 Proof of Lemma 4.1

**Lemma 10.1.** Let \( x \) and \( y \) be two irrelevant nodes in an isomorphically ordered tree. A unique node \( w \) and unique indices \( i \) and \( j \), \( i \neq j \), exist such that \( x \) belongs to \( T(c_i(w)) \) and \( y \) belongs to \( T(c_j(w)) \).

**Proof.** Let \( u_1, \ldots, u_{m(x)} \) and \( v_1, \ldots, v_{m(y)} \) correspond to the paths from the root of the tree to \( x \) and \( y \) respectively. Clearly \( u_1 = v_1 \) is the root, but the two sequences diverge later. Therefore, there is a unique index \( x \) such that \( u_x = v_x \) but \( u_{x+1} \neq v_{x+1} \). Clearly, \( u_{x+1} \) and \( v_{x+1} \) are children of \( u_x \) in the tree. In addition, \( x \) resides in \( T(u_{x+1}) \) while \( y \) resides in \( T(v_{x+1}) \).

Now back to the proof of Lemma 4.1, if \( x \) and \( y \) do not belong to the same GOP, then the lemma clearly holds. Now suppose they belong to the same GOP. Based on Lemma 10.1, there exist a node \( w \) and indices \( i \) and \( j \) such that \( x \) belongs to \( T(c_i(w)) \) and \( y \) belongs to \( T(c_j(w)) \). If \( i < j \), then the definition of isomorphically ordered trees gives \( \max(T(c_i(w))) \leq \min(T(c_j(w))) \), but since none of the playback deadlines are equal, we have \( \max(T(x)) < \min(T(y)) \). If \( i > j \), it can be shown similarly \( \min(T(x)) > \max(T(y)) \) which is not possible due to \( d_x < d_y \). As a result, \( i < j \) and \( \max(T(x)) < \min(T(y)) \).

10.3 Hierarchical Dyadic Structure

We first present a mathematical characterization of the hierarchical dyadic structure and then show that it belongs to the quasi-SIO class.

In the hierarchical dyadic structure \( GnBm \), the GOP size \( n \) is an integer power of 2, and \( m \) is the number of B-frames between consecutive non-B-frames, with \( m = 2^\omega - 1 \) for some \( \omega \in \mathbb{N} \). Each GOP contains one leading I-frame and \( \frac{n}{m+1} - 1 \) P-frames. Each P-frame depends on the previous I-frame/P-frame in the display order. The dependency structure among the B-frames between two consecutive non-B-frames is described by Dyadic-build\((i,j)\) in Algorithm 3, where \( j - i \) is an integer power of 2. Note that in this algorithm, the frames of the video sequence are indexed in the display order as integers.

**Algorithm 3** Dyadic-build\((i,j)\)

1. if \( |i - j| \leq 1 \) or \( \log_2 |i - j| \notin \mathbb{N} \) then
2. return
3. end if
4. \( i_0 \leftarrow \frac{i+j}{2} \)
5. \( i_0 \) depends on \( i, j \)
6. Dyadic-build\((i_0,i)\)
7. Dyadic-build\((i_0,j)\)
As an example, Figure 3.1 shows the hierarchical dyadic structure for a GOP with $G\!16B3$. A sequence of frames $i, \ldots, j$ is called a complete sequence if, except $i$ and $j$, none of the frames in the sequence have parents outside this sequence. Thus, for a complete sequence, if the frames $i$ and $j$ are available, all frames in the sequence can be decoded. An example of a complete sequence is a GOP. It can easily be shown through induction that, in the context of the hierarchical dyadic structure, the size of any complete sequence is $2^{\omega'} + 1$, for some $\omega' \in \mathbb{N}$.

The hierarchical dyadic structure does not belong to the SIO class. This is because there is always a B-frame that is the descendant of two consecutive I-frames, which violates the sequential property. In Figure 3.1, frame 14 is an example of such a frame that violates the sequential property.

However, in what follows, we show that the hierarchical dyadic structure belongs to the quasi-SIO class. This is accomplished by demonstrating that a modification on the hierarchical dyadic structure, which removes the backward DAG edges emanating from each I-frame to its children in the preceding GOP, results in a structure belonging to the SIO class. In this modified hierarchical dyadic structure, no B-frame depends on a succeeding I-frame. For instance, this means in Figure 3.1, frames 14 and 15 no longer depend on frame 16. It is worth mentioning that this modification is used as an approximation in [18]. However, in our work, we show that an optimal schedule can be obtained for the hierarchical dyadic structure without modification, since it is a special case of the quasi-SIO class.

**Theorem 10.1.** The DAG of a modified hierarchical dyadic structure is sequential.

*Proof.* The MBFS tree corresponding to each GOP is always rooted at the GOP’s leading I-frame. If all B-frames are removed, only some subtrees of the MBFS forest are removed, and the resultant DAG consisting of only I-frames and P-frames is clearly sequential. Therefore, it suffices to only prove that the sequential property is preserved when the MBFS algorithm is executed on B-frames between consecutive non-B-frames.

We use mathematical induction. In particular, since the number of B-frames between two consecutive non-B-frames is $2^\omega - 1$, for some $\omega \in \mathbb{N}$, induction is carried out on $\omega$. In the following, we address two cases based on whether the latter non-B-frame is 1) an I-frame or 2) a P-frame.

For the basis step, consider $\omega = 1$. This situation is depicted in Figures 10.3(a) and 10.3(b) for cases 1) and 2) respectively. Note that in case 1), since frame $z$ is an I-frame, the edge from $z$ to $y$ is removed in the modified hierarchical dyadic structure. In both cases, the MBFS tree clearly satisfies the sequential property.

For the induction step, suppose the statement holds for $\omega = k$. This is shown in Figures 10.4(a) and 10.4(b) for cases 1) and 2) respectively. In both cases, the number of frames between the two end frames is $2^k - 1$, and the resulting tree satisfies the sequential property. The triangles in both figures indicate
Figure 10.4: Hierarchical dyadic structure with $\omega = k$ used in the proof of Theorem 10.1. $x$ and $z$ are non-B-frames, and all frames between them are B-frames.

The subtree rooted at node $y$. We consider $\omega = k + 1$ for the two cases separately.

Case 1):

Figure 10.5(a) depicts this situation. There are $2^{k+1} - 1$ nodes between $x$ and $w$. Node $z$ is located halfway between them. In addition, node $y$ is halfway between $x$ and $z$, and node $u$ is halfway between $z$ and $w$. The dependency structure among these indicated nodes is displayed, and the rest of the dependency structure, which is not shown, is determined by the modified hierarchical dyadic structure.

The MBFS algorithm first visits $x$, then to $z$, $y$, and finally $u$. It can be observed that the dependency structure for frames between $x$ and $z$ is the same as Figure 10.4(b), and the dependency structure for frames between $z$ and $w$ is the same as Figure 10.4(a). Hence, the triangles in Figure 10.5(a) refer to the same triangles as in Figures 10.4(a) and 10.4(b).

For nodes $x$, $y$, $z$, and $u$, it is easy to see that the sequential property holds. Within the subtree rooted at $y$, due to the inductive hypothesis in Figure 10.4(b), the sequential property is satisfied. Now consider the additional ancestors $x$ and $z$. All paths from $x$ to nodes in the subtree rooted at $y$ pass through $z$ and $y$. Therefore for all nodes in the subtree rooted at $y$, the sequential property is preserved. Similarly, the same can be shown for the subtree rooted at $u$.

Case 2):

Figure 10.5(b) depicts this situation. Similar to the previous case, again there are $2^{k+1} - 1$ nodes between nodes $x$ and $w$. Node $z$ is located halfway between them. Node $y$ is halfway between $x$ and $z$, and node $u$ is halfway between $z$ and $w$. The dependency structure among these indicated nodes is displayed, and the rest of the dependency structure, which is not shown, is determined by the modified hierarchical dyadic structure.

The MBFS algorithm first visits $x$. Different from the previous case, however, it then visits $w$, before moving to $z$, $y$, and finally $u$. When it visits $z$, nodes $x$ and $w$ have already been visited, so the remaining two branches of the dependency structure, from $z$ to $y$ and from $z$ to $u$, each resembles that of 10.4(a). In particular, $z$ in Figure 10.5(b) plays the role of $x$ in Figure 10.4(a), while $y$ and $u$ in Figure 10.5(b) plays the role of $y$ in Figure 10.4(a).

Similar to the previous case, for nodes $x$, $w$, $z$, $y$, and $u$, it is easy to see that the sequential property holds. Within the subtree rooted at $y$, due to the inductive hypothesis in Figure 10.4(b), the sequential property is satisfied. Now consider the additional ancestors $x$, $w$, and $z$. All paths from $x$ to nodes in the subtree rooted at $y$ pass through $w$, $z$, and $y$. Therefore for all nodes in the subtree rooted at $y$, the sequential property is preserved. Similarly, the same holds for the subtree rooted at $u$. □
Figure 10.5: Hierarchical dyadic structure with $\omega = k + 1$ used in the proof of Theorem 10.1. $x$ and $w$ are non-B-frames, and all frames between them are B-frames.
**Theorem 10.2.** The MBFS trees under the hierarchical dyadic structure (modified or not) are Binary Search Trees (BSTs) with respect to the display deadlines.

**Proof.** Consider a hierarchical dyadic GOP with $G_{nBm}$. To show that its corresponding MBFS tree is a BST, we need to verify the following three properties for every node in a BST: the node has at most two children, the subtree rooted at its left child contains only nodes with deadlines earlier than that of the node, and the subtree rooted at its right child contains only nodes with deadlines later than that of the node.

Since in the creation of each MBFS tree, the dependencies on the succeeding I-frame does not apply, in what follows, we only need to consider a modified version of Algorithm 3 that omits such dependencies. Assume the video sequence is sorted and indexed in display order by integers. We next consider each node in the DAG $G$ and show that when the MBFS algorithm visits the node, the BST properties are satisfied. This is carried out for each type of nodes in the following.

- **I-frame:** The I-frame is always the root of its own MBFS tree. If $m < n - 1$, then there exists at least one P-frame in the GOP. After the MBFS algorithm visits this I-frame, among all of its children, only the first P-frame in display order is decodable, since the other children of the I-frame are also descendants of the first P-frame. Hence, the I-frame has only a single child with respect to the MBFS tree, so the BST properties are satisfied.

  If $m = n - 1$, then the B-frame in the mid point of the GOP depends only on this I-frame in the modified structure, so it acts as a P-frame. Similarly, this B-frame is the only child of the I-frame with respect to the MBFS tree, so the BST properties are satisfied.

- **P-frame:** Let the considered P-frame be $i_0$. The preceding and succeeding non-B-frames are the frames labelled $i_0 - (m + 1)$ and $i_0 + (m + 1)$ respectively. All children of $i_0$ are between these two frames.

  Note that the subsequence $i_0 - (m + 1), \ldots, i_0$ is a complete sequence. Among these frames, after the MBFS algorithm visits $i_0$, the only decodable child of $i_0$ is $\frac{i_0 - (m + 1) + i_0}{2}$, since the other children of $i_0$ are all descendants of $\frac{i_0 - (m + 1) + i_0}{2}$. The subsequence $i_0, \ldots, i_0 + (m + 1)$ is also a complete sequence, but we need to consider two cases depending on whether frame $i_0 + (m + 1)$ is an I-frame or a P-frame. If $i_0 + (m + 1)$ is an I-frame, then after the MBFS algorithm visits $i_0$, the only decodable child of $i_0$ among $i_0, \ldots, i_0 + (m + 1)$ is $\frac{i_0 + (i_0 + (m + 1))}{2}$, since the other children all depend on $\frac{i_0 + (i_0 + (m + 1))}{2}$. If $i_0 + (m + 1)$ is a P-frame, then after the MBFS algorithm visits $i_0$, the only decodable child of $i_0$ among $i_0, \ldots, i_0 + (m + 1)$ is $i_0 + (m + 1)$, since the other children all depend on $i_0 + (m + 1)$. In both cases, $i_0$ has only one decodable child among $i_0, \ldots, i_0 + (m + 1)$.

  As a result, $i_0$ has exact two children in the MBFS tree. Moreover, the sequences $i_0 - (m + 1), \ldots, i_0 - 1$ and $i_0 + 1, \ldots, i_0 + (m + 1)$ are completely irrelevant, and all of the former have deadlines earlier than $d_{i_0}$ and all of the latter have deadlines later than $d_{i_0}$. Therefore, all BST properties are satisfied.

- **B-frame:** Let the considered B-frame be $i_0$. This is similar to the previous case, except that instead of frames between $i_0 - (m + 1)$ and $i_0 + (m + 1)$, we should consider frames $i_0 - (m_0 + 1), \ldots, i_0 + (m_0 + 1)$ for some $m_0$ such that $m_0 < \frac{m + 1}{2}$ and $i_0 - (m_0 + 1), \ldots, i_0 + (m_0 + 1)$ is the largest complete sequence centered at $i_0$. We similarly see that, after the MBFS algorithm visits $i_0$, the
only decodable children of \( i_0 \) are \( \frac{(i_0 - (m_0 + 1)) + i_0}{2} \) and \( \frac{i_0 + (m_0 + 1)}{2} \). The other properties of BST are also similarly satisfied.

The BSTs are a special case of isomorphically ordered trees. Combining this with Theorem 10.1, we see that the modified hierarchical dyadic structure belongs to the SIO class, and hence, the hierarchical dyadic structure belongs to the quasi-SIO class.

### 10.4 Proof of Theorem 4.1

Given a transmission sequence that does not necessarily meet the two properties, it will be modified to meet those properties in two steps.

**First Step:** With the assumption that the first property does not hold, there must be frames \( l_1 \) and \( l_2 \) such that frame \( l_1 \) is the ancestor of frame \( l_2 \) but \( l_2 \) is scheduled before \( l_1 \). An example of this is shown in Figure 10.6(a). Let \( K_{l_2} = \{ \text{descendants of } l_2 \text{ scheduled after } l_2 \text{ and before } l_1 \} \). Also let the index of the timeslot in which the transmission of \( l_2 \) starts be \( e_1 \) and the index of the timeslot in which the transmission of \( l_1 \) ends be \( e_2 \). Recall at least one member of the set \( K_{l_2} \) has to be successful; otherwise, the entire set can be dropped with no effect on the reward function. In addition, the transmission of frame \( l_1 \) has to terminate prior to the deadline of its successful descendants. As a result:

\[
e_2 \leq d_l, \forall l \in K_{l_2} \cup \{ l_2 \}, \ l \text{ is successful} \quad (10.1)
\]

Now consider the frames in the interval \( [e_1, e_2] \). They can be divided into three sets: \( \{ l_1 \}, A_1 = K_{l_2} \cup \{ l_2 \} \), and \( A_2 = \{ \text{All the frames in interval } [e_1, e_2] \} - (A_1 \cup \{ l_1 \}) \). Now starting from timeslot \( e_1 \) first schedule \( A_2 \), then \( l_1 \), and then \( A_1 \), such that the order of frames in \( A_2 \) in the original sequence is preserved and also the same for \( A_1 \). An example of this operation is shown in Figure 10.6(b).

Any previously successful frame in the time interval \( [e_1, e_2] \) remains successful. In particular, since all frames in \( A_2 \) either move backward in time or do not change position, the previously successful ones remain successful. Moreover, due to inequality (10.1) all previously successful frames in \( A_1 \) remain successful as well. Furthermore, no new violation of the first property is created. Since this operation always repairs at least one pair that violates the first property, and the number violations is finite, repetition of the operation will end in an solution which satisfies the first property.

**Second Step:** Let \( U \) be the set of frames scheduled between \( l_j \) and \( l_i \). Suppose all of them are irrelevant with respect to \( l_j \). Let \( t_0 \) be the timeslot when the transmission of \( l_j \) starts and \( t_1 \) be the timeslot when the transmission of \( l_i \) ends. If \( l_i \) is successful, then \( t_1 \leq d_{l_i} < d_{l_j} \).

Else, if \( l_i \) is not successful, then it has to have a successful descendant, namely \( f_i \). Let \( t_2 \) be the timeslot when the transmission of \( f_i \) ends. Since \( f_i \) is successful: \( t_2 \leq d_{f_i} \). Furthermore, from the
previous step, we have \( t_1 < t_2 \). We next consider two cases based on whether \( l_i \) is an I-frame. If \( l_i \) is not an I-frame, then \( f_i \) belongs to the same tree as \( l_i \) in the MBFS forest. Hence, Lemma 4.1 indicates \( d_{f_i} < d_{l_j} \). If \( l_i \) is an I-frame, let \( n_l \) be the index of the GOP that \( l_i \) belongs to. Then since \( d_{l_1} < d_{l_j} \), the frame \( l_j \) has to belong to a GOP with an index higher than \( n_l \). But, the children of \( l_i \) are only in \( GOP_{n_l-1} \) and \( GOP_{n_l} \). As a result, \( d_{f_i} < d_{l_j} \). Hence, in both cases, we have \( t_1 < t_2 < d_{l_j} \).

Now, we can re-order the frames starting at \( t_0 \) as follows: \( U \), then \( l_i \), and then \( l_j \). This clearly will not harm \( l_i \) and \( l_j \). In addition, the frames of \( U \) are moved backward in time, so none of them is harmed either. Furthermore, none of the frames outside of \([t_0, t_1]\) are affected. This operation is illustrated in Figure 10.7. A special case of this is when \( U \) is empty, i.e., when \( l_j \) and \( l_i \) are neighbors.

### 10.5 Proof of Lemma 4.2

We use mathematical induction to prove the first part.

For the basis step, consider \( k = 1 \). Since \( o_1 \) and \( f_1 \) are from different GOPs and they are irrelevant, due to the second property of Theorem 4.1, \( d_{f_1} < d_{o_1} \) and \( o_1 \) belongs to a GOP with an index greater than \( i \).

For the induction step, suppose the statement holds for \( k = m \). That is, \( d_{f_1} < d_{o_m} \) and \( o_m \) belongs to some \( GOP_j \) with \( j > i \). The frame \( o_{m+1} \) either is a descendant of \( o_m \) or is irrelevant with respect to \( o_m \).

If \( o_{m+1} \) is a descendant of \( o_m \), then it belongs to \( GOP_j \) or \( GOP_{j-1} \). In the former case, we have \( j > i \) and hence \( d_{f_1} < d_{o_{m+1}} \). In the latter case, since \( o_{m+1} \) and \( f_1 \) do not belong to the same GOP, \( j - 1 \neq i \). Combining this with \( j > i \), we have \( j - 1 > i \) and hence \( d_{f_1} < d_{o_{m+1}} \).

If \( o_{m+1} \) is irrelevant with respect to \( o_m \), due to the second property of Theorem 4.1, \( d_{o_m} < d_{o_{m+1}} \) and \( o_{m+1} \) cannot belong to a GOP with an index less than \( j \). As a result, \( d_{f_1} < d_{o_{m+1}} \) and \( o_{m+1} \) belongs to a GOP with an index larger than \( i \).

The second part can be similarly proven using mathematical induction.

### 10.6 Proof of Lemma 4.3

We prove the first statement using mathematical induction.

For the basis step, consider \( k = 1 \). Since \( o_1 \) is scheduled after \( f_1 \) and they are irrelevant, due to the second property of Theorem 4.1, we have \( d_{f_1} < d_{o_1} \).

For the induction step, suppose the statement holds for \( k = m \). Consider the case \( k = m + 1 \). We first note that none of \( o_1, \ldots, o_k \) can be an I-frame, since otherwise \( f_1 \) would be scheduled after the I-frame. The inductive hypothesis says \( d_{f_1} < d_{o_m} \). The frame \( o_{m+1} \) either is a descendant of \( o_m \) or is irrelevant with respect to \( o_m \). If it is a descendant, then according to Lemma 4.1, \( d_{f_1} < d_{o_{m+1}} \). If \( o_{m+1} \) is irrelevant with respect to \( o_m \), due to the second property of Theorem 4.1, \( d_{o_m} < d_{o_{m+1}} \). As a result, \( d_{f_1} < d_{o_{m+1}} \).
The second statement can be proved similarly.

### 10.7 Proof of Lemma 5.2

Assume towards a contradiction that the lemma does not hold. Then it can be shown easily that frames $l'$, $l_1$, and $l_2$ exist such that $l_1, l_2 \in T(l')$ and none of the frames scheduled between $l_1$ and $l_2$ belong to $T(l')$.

Without loss of generality, suppose $l_1$ is scheduled earlier than $l_2$. Index the frames scheduled between $l_1$ and $l_2$ in the ascending order of transmission starting time to obtain $j_1, \ldots, j_{k'}$ for some integer $k'$. These frames are irrelevant with respect to $l'$ and therefore irrelevant with respect to $l_1$ and $l_2$. Hence, according to Lemma 4.3,

$$d_{l_1} < d_{j_m} < d_{l_2}, 1 \leq m \leq k'. \tag{10.2}$$

Now consider any specific $j_m$ among $j_1, \ldots, j_{k'}$. Lemma 4.1 suggests that either $\max(T(l')) < \min(T(j_m))$ or $\min(T(l')) > \max(T(j_m))$ depending on the relative sizes of $d_{j_m}$ and $d_{l'}$. In the former case, since $l_1$ and $l_2$ belong to $T(l')$, both $d_{l_1}$ and $d_{l_2}$ are smaller than $d_{j_m}$, while in the latter case, similarly both $d_{l_1}$ and $d_{l_2}$ are greater than $d_{j_m}$. This contradicts 10.2.

### 10.8 Proof of Lemma 6.5

Towards a contradiction, let $o_1, \ldots, o_k$ be the sequence of frames scheduled between $I_{i+1}$ and $\zeta_i$. These frames belong to $N_i$ based on Lemma 6.3. Frame $o_1$ cannot be in $N_i - D_i$, since otherwise the second property of Theorem 4.1 would be violated, due to $d_{o_1} < d_{I_{i+1}}$. Hence, $o_1 \in D_i$, and $o_1$ is a critical node. Then by the definition of $\zeta_i$, $d_{\zeta_i} < d_{o_1}$, and $\zeta_i$ and $o_1$ are irrelevant. Furthermore, by Theorem 4.1, each frame $o_k$, $k > 1$, is either the descendant of $o_{k-1}$ or irrelevant with respect to $o_{k-1}$, and $d_{o_{k-1}} < d_{o_k}$. Hence, the frames $o_2, \ldots, o_k$ are either descendants of $o_1$, or irrelevant with respect to $o_1$ but have a display deadline greater than $d_{o_1}$. Frames in the latter case cannot be ancestors of $\zeta_i$, since otherwise according to Lemma 4.1, $d_{o_k} < d_{\zeta_i}$, which is a contradiction. Therefore, $o_2, \ldots, o_k$ are irrelevant with respect to $\zeta_i$. In addition, based on Lemma 4.1, all of $o_1, \ldots, o_k$, especially $o_k$, have a deadline greater than $d_{\zeta_i}$. However, since $o_k$ and $\zeta_i$ are neighbors, this violates the second property of Theorem 4.1.

### 10.9 Proof of Lemma 6.6

For the first statement, assume towards a contradiction that the statement does not hold, i.e., there exists some frame in $N_i - D_i$, such that it is irrelevant with respect to $\zeta_i$ and has a deadline earlier than $d_{\zeta_i}$, and it is scheduled after $I_{i+1}$. Let $x$ be the one of such frames that has the earliest scheduled transmission start time. Then, by definition $d_x < d_{\zeta_i}$. Furthermore, let $y$ be the frame scheduled immediately prior to $x$. As explained previously, Lemmas 6.1, 6.2, 6.3, and 6.4 jointly dictate that in the quasi-SIO canonical schedule all frames in $N_i \cup \{I_{i+1}\}$ must be sent together. Hence, either $y \in N_i$ or $y = I_{i+1}$. However, since $x \neq \zeta_i$, by Lemma 6.5, $y$ cannot be $I_{i+1}$. Therefore, $y$ is scheduled between $I_{i+1}$ and $x$, and only the following two cases are left for $y$:

1. $y \in N_i - D_i$
In this case, since \( y \notin \mathcal{D}_i \) clearly \( y \) is not \( \zeta_i \) nor a descendant of \( \zeta_i \). Furthermore, by the first property of Theorem 4.1 and Lemma 6.5, all ancestors of \( \zeta_i \) other than \( I_{i+1} \) must be scheduled ahead of \( I_{i+1} \). Therefore, \( y \) is not an ancestor of \( \zeta_i \). Hence, \( y \) is irrelevant with respect to \( \zeta_i \). Furthermore, we have \( d_{\zeta_i} < d_y \), since otherwise the existence of \( y \) would contradict the definition of \( x \).

This leads to two observations. First, \( d_x < d_y \), which implies that \( x \) and \( y \) cannot be irrelevant, due to the second property of Theorem 4.1. Second, Lemma 4.1 suggests that \( \max(T(\zeta_i)) < \min(T(y)) \). This implies that \( y \) cannot be an ancestor of \( x \), since otherwise we would have \( d_{\zeta_i} < d_x \), contradicting the definition of \( x \).

Finally, the first property of Theorem 4.1 suggests that \( y \) is not a descendant of \( x \), since \( y \) is schedule ahead of \( x \). Hence, this case is not possible.

2. \( y \in \mathcal{D}_i \)

We first note that \( x \) cannot be a descendant of \( y \) since \( x \notin \mathcal{D}_i \), and \( x \) cannot be an ancestor of \( y \) due to the first property of Theorem 4.1. Hence, \( x \) and \( y \) are irrelevant.

Let \( z \) be the critical frame such that \( y \) belongs to \( T(z) \) (with possibly \( y = z \)). If \( z = \zeta_i \), then since \( x \) and \( \zeta_i \) are irrelevant, Lemma 4.1 indicates \( \max(T(x)) < \min(T(z)) \), so we have \( d_x < d_y \). If \( z \neq \zeta_i \), then \( z \) and \( \zeta_i \) are irrelevant and \( d_{\zeta_i} < d_z \); therefore Lemma 4.1 indicates \( d_{\zeta_i} < d_y \). Hence, in both cases, \( d_x < d_y \). This violates the second property of Theorem 4.1.

For the second statement, assume towards a contradiction that the statement does not hold, i.e., there exists some frame in \( \mathcal{N}_i - \mathcal{D}_i \), such that it is irrelevant with respect to \( \zeta_i \) and has a deadline later than \( d_{\zeta_i} \), and it is scheduled before \( I_{i+1} \). Let \( x \) be the one of such frames that has the latest scheduled transmission start time. Then, by definition \( d_{\zeta_i} < d_x \). Let \( y \) be the frame scheduled immediately after \( x \). As explained previously, Lemmas 6.1, 6.2, 6.3, and 6.4 jointly dictate that in the quasi-SIO canonical schedule all frames in \( \mathcal{N}_i \cup \{I_{i+1}\} \) must be sent together. Hence, we have \( y \in \mathcal{N}_i \) or \( y = I_{i+1} \).

1. \( y \in \mathcal{N}_i \)

In this case, since \( y \) is scheduled before \( I_{i+1} \), the first property of Theorem 4.1 suggests that \( y \notin \mathcal{D}_i \). This implies that \( y \) cannot be \( \zeta_i \) or a descendant of \( \zeta_i \).

Now, suppose \( y \) is an ancestor of \( \zeta_i \). Then, \( x \) cannot be an ancestor of \( y \), since otherwise \( x \) would not be irrelevant with \( \zeta_i \). However, \( x \) cannot be a descendant of \( y \) either, due to the first property of Theorem 4.1. Therefore, \( x \) and \( y \) are irrelevant. Then, since \( x \) and \( y \) are scheduled next to each other, the second property of Theorem 4.1 suggests that \( d_x < d_y \), which implies \( \max(T(x)) < \min(T(y)) \) by Lemma 4.1. However, this also implies that \( d_{\zeta_i} < d_y \), which contradicts the definition of \( x \). Hence, we conclude that \( y \) cannot be an ancestor of \( \zeta_i \).

The above implies that \( y \) and \( \zeta_i \) are irrelevant. Hence, we have \( d_y < d_{\zeta_i} \), since otherwise the existence of \( y \) would contradict the definition of \( x \). This leads to two observations. First, \( d_y < d_x \), which implies that \( x \) and \( y \) cannot be irrelevant, due to the second property of Theorem 4.1. Second, \( y \) cannot be a descendant of \( x \). This is because \( \max(T(\zeta_i)) < \min(T(x)) \) by Lemma 4.1, so we would have \( d_{\zeta_i} < d_y \) if \( y \) were a descendant of \( x \).

Furthermore, by the first property of Theorem 4.1, \( y \) cannot be an ancestor of \( x \), since \( y \) is schedule after \( x \). Hence, this case is not possible.
2. \( y = I_{i+1} \)

In this case, Lemma 6.5 suggests that \( y = I_{i+1} \) is followed immediately by \( \zeta_i \). Hence, we have the subsequence \( xI_{i+1}\zeta_i \), with \( d_{\zeta_i} < d_x \). Furthermore, since \( x \in N_i - D_i \), it is irrelevant with respect to \( I_{i+1} \). This violates the second property of Theorem 4.1.

### 10.10 Proof of Theorem 7.1

The proof consists of two parts: 1-Proving the problem is NP 2- Polynomially reducing a known NP-complete problem to this problem.

For the first part, it has to be shown a sample solution can be verified in polynomial time. Given any transmission sequence as a schedulable subsequence, clearly the transmission start and end times can be determined and checked with the display in polynomial time.

For the second part, the Knapsack problem is reduced to the Generalized Scheduling Problem. From [35], the Knapsack problem is known to be NP-complete. The Knapsack problem deals with placing objects in a Knapsack. The Knapsack has a specified volume capacity and each object has a value; the objective is to maximize the some of the values of the objects placed in the Knapsack. The problem reduction is performed as follows:

The channel capacity is set equal to unity. Each object is considered as a frame. Its size is set equal to its volume. Moreover, the distortion value of each object is set to its value. Arbitrarily put the frames in a sequence and index them to obtain \( \{f_i\} \). Let \( f_i \) be the single frame of \( GOP_i \). All frames will act as I-frames. Clearly, if \( i < j \), then \( d_x \leq d_y \), \( \forall x \in GOP_i, \forall y \in GOP_j \). The Knapsack capacity is assigned as the deadline to all the DUs. Clearly the Knapsack problem has a solution if and only if this problem has a solution.

Therefore, the Scheduling Problem is NP-complete.

### 10.11 Proof of Lemma 7.1

Let \( u_1, \ldots, u_m(x) \) and \( v_1, \ldots, v_m(y) \) correspond to the paths from the root of the tree to \( x \) and \( y \) respectively. Clearly \( u_1 = v_1 \) is the root, but the two sequences diverge later. Therefore, there is a unique index \( x \) such that \( u_x = v_x \) but \( u_{x+1} \neq v_{x+1} \). Clearly, \( u_{x+1} \) and \( v_{x+1} \) are children of \( u_x \) in the tree. In addition, \( x \) resides in \( T(u_{x+1}) \) while \( y \) resides in \( T(v_{x+1}) \).

### 10.12 Proof of Lemma 7.2

Let \( l_1, l_2 \) and \( l_3 \) be three frame in an MBFS tree. We have to prove that if \( l_1C_2l_2 \) and \( l_2C_3l_3 \), then \( l_1C_3l_3 \). For that purpose, it will be shown first that \( l_1 \) and \( l_3 \) satisfy the second conditions of Definition 7.1. This condition implies \( l_1 \) and \( l_3 \) are irrelevant. So, we will have \( l_1C_3l_3 \).

Since \( l_1C_2l_2 \), the second condition of definition 7.1 indicates there are a frame \( w(l_1, l_2) \) and indices \( i, j \) (\( i < j \)) such that \( l_1 \) belongs to \( T(c_i(w(l_1, l_2))) \) and \( l_2 \) belongs to \( T(c_j(w(l_1, l_2))) \). Similarly, since \( l_1C_2l_2 \), there are a frame \( w(l_2, l_3) \) and indices \( k, l \) (\( k < l \)) such that \( l_2 \) belongs to \( T(c_k(w(l_2, l_3))) \) and \( l_3 \) belongs to \( T(c_l(w(l_2, l_3))) \).

Now the sequential property of the MBFS forest implies \( w(l_1, l_2) \) and \( w(l_2, l_3) \) can not be irrelevant, since both are the ancestors of \( l_2 \). If \( w(l_1, l_2) \) is the ancestor of \( w(l_2, l_3) \), then \( w(l_2, l_3) \) has to lie in...
$T(c_j(w(l_1,l_2)))$. As a result, also $l_3$ has to lie in $T(c_j(w(l_1,l_2)))$, since it is a descendant of $w(l_2,l_3)$. Therefore, $l_1$ and $l_3$ satisfy the second condition of definition 7.1 and $l_1Cl_3$. If $w(l_2,l_3)$ is the ancestor of $w(l_1,l_2)$, then it can be shown similarly that $w(l_1,l_2)$ has to lie in $T(c_k(w(l_2,l_3)))$. Hence, $l_1$ and $l_3$ again satisfy the second condition of definition 7.1 and $l_1Cl_3$.

10.13 Proof of Lemma 7.3

Let $n_1$, $n_2$ and $n_3$ be the index of the GOPs containing $l_1$, $l_2$ and $l_3$ respectively. Since $l_1$ and $l_3$ are irrelevant, it suffices to show that one of the conditions of definition 7.1 holds. Four case have to be considered:

1. $n_1 \neq n_2$, $n_2 \neq n_3$
   Since $l_1Cl_2$, only the first condition of definition 7.1 applies; so, $n_1 < n_2$. Similarly, since $l_2Cl_3$, $n_2 < n_3$. As a result, $n_1 < n_3$. Hence, the first condition of definition 7.1 holds and $l_1Cl_3$.

2. $n_1 = n_2$, $n_2 \neq n_3$
   Similar to the previous case, since $l_2Cl_3$, $n_2 < n_3$. As a result, $n_1 < n_3$. So, the first condition of definition 7.1 holds and $l_1Cl_3$.

3. $n_1 \neq n_2$, $n_2 = n_3$
   Similar to the first case, since $l_1Cl_2$, so, $n_1 < n_2$. As a result, $n_1 < n_3$. So, the first condition of definition 7.1 holds and $l_1Cl_3$.

4. $n_1 = n_2 = n_3$
   Already shown in the proof of lemma 7.2 that $l_1Cl_3$.

10.14 Proof of Lemma 7.4

If $x$ and $y$ do not belong to the same GOP, then the lemma clearly holds. Now suppose they belong to the same GOP. Based on Lemma 7.1, there exist a node $w = w(x,y)$ and indices $i$ and $j$ such that $i < j$ and $x$ belongs to $T(c_i(w))$ and $y$ belongs to $T(c_j(w))$. For all nodes $l_1$ and $l_2$ in $T(x)$ and $T(y)$ respectively, it is easy to verify $w(x,y) = w(l_1,l_2)$; therefore, $l_1Cl_2$. Furthermore, the definition of isomorphically ordered trees gives $\max(T(c_i(w))) \leq \min(T(c_j(w)))$. As a result, we have $\max(T(x)) \leq \min(T(y))$. 
Bibliography


