Network Source Coding: Theory and Code Design for Broadcast and Multiple Access Networks

Thesis by
Qian Zhao

In Partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy

California Institute of Technology
Pasadena, California
2003
(Defended May 19th, 2003)

To my parents

and

Hanying

Acknowledgments

I wish to express my enormous gratitude to my advisor, Professor Michelle Effros of the Data Compression Lab at California Institute of Technology. Her support, guidance and encouragement throughout the period of my graduate studies made the completion of this work possible.

Professors Yaser S. Abu-Mostafa, Jehoshua (Shuki) Bruck, Michelle Effros, Steven Low, and Robert J. McEliece served on my Ph.D. examination committee and were very careful in reviewing my thesis manuscript. This thesis benefits enormously from their valuable comments. I also want to thank Professor Tom Hou for his consistent support in my study.

I wish to thank the most significant persons in my life: my parents and my husband Hanying Feng for their unconditional love and support for me.

I would also like to thank my former and current fellow graduate students in the Communications Group of Caltech for their friendship and help, especially Diego Dugatkin, Michael Fleming, Sidharth Jaggi, Hui Jin, Lifang Li, Mostafa Elkhamy, and Chaitanya Rao.

I would like to thank Mr. Howard Oringer for his consistent support of me and of the Data Compression Lab.

This material is based upon work partially supported by NSF Grant No. CCR-9909026,

NSF Grant No. CCR-0220039, Caltech's Lee Center for Advanced Networking, Howard Oringer Scholarship and Chinese-American Engineers and Scientist Association of Southern California Scholarship.

Abstract

In the information age, network systems and applications have been growing rapidly to provide us with more versatile and high bit rate services. However, the limited bandwidth restricts the amount of information that can be sent through the networks. Thus efficient data representation or source coding is imperative for future network development. Distinct from the traditional source coding strategy, *network source codes* take advantage of the network topology and are able to maximally compress data before transmission.

In this thesis, I present a variety of source coding techniques for use in network environments and demonstrate the benefits of network source codes over traditional source codes from both theoretical and practical perspectives.

First, I address source coding for broadcast systems. The results I obtain include derivation of the theoretical limits of broadcast system source codes, algorithm design for optimal broadcast system vector quantizers, implementation of the optimal code, and experimental results.

Then, I focus on multiple access systems which are the dual systems of broadcast systems. I present the properties of multiple access source codes and generalize traditional entropy code design algorithms to attain the corresponding optimal multiple access source codes for

arbitrary joint source statistics. I further introduce a family of polynomial complexity code design algorithms that approximates the optimal solutions. Application to universal coding for multiple access networks when the joint source statistics are unknown *a priori* is briefly discussed. Finally, I demonstrate algorithmic performance by showing experimental results on a variety of data sets.

Finally, in seeking a simple lossy source coding method for general networks, I apply entropy constrained dithered quantization in network source code design and present the coding results for multi-resolution source codes and multiple access source codes. Multi-resolution and multiple access dithered quantizers are low complexity codes that achieve performance very close to the theoretical rate-distortion bound.

Contents

Acknowledgments **iv**

Abstract **vi**

1 Introduction **1**

2 Source Coding for Broadcast Systems **5**

2.1 Introduction 5

2.2 Notation 9

2.3 Theoretical Bounds on Lossless Codes 12

2.4 Lossy Code Design 17

2.5 Experimental results 19

2.6 Summary 22

3 Source Coding for Multiple Access Systems **23**

3.1 Introduction 23

3.2 Lossless Instantaneous Side Information Source Codes 26

3.3 Lossless Instantaneous Multiple Access Source Codes 49

3.4 Near-Lossless Instantaneous Multiple Access Source Codes 55

3.5 Low Complexity Multiple Access Source Coding Algorithms 57

3.6 Uniquely Decodable Multiple Access Source Codes 66

3.7 Experimental Results 81

3.8 Summary 94

4 Entropy Constrained Dithered Quantization **97**

4.1 Introduction 97

4.2 Multi-resolution Source Codes 101

4.3 Multiple Access Source Codes 112

4.4 Summary 117

5 Summary **120**

6 Appendix **123**

Bibliography **130**

List of Figures

2.1 Traditional (single-encoder single-decoder) source codes for broadcast system. 7

2.2 Broadcast System Source Coding. 7

2.3 The k-ary tree structure for 2-receiver broadcast system lossy code. 17

2.4 Fixed-rate BSVQ vs. fixed-rate VQ. 21

2.5 Variable-rate BSVQ vs. ECVQ. 21

2.6 Variable-rate BSVQ vs. fixed-rate BSVQ. 22

3.1 (a) An MASC and (b) the Slepian-Wolf achievable rate region for MASCs. . 24

3.2 (a) Partition tree $\mathcal{T}(\mathcal{P}(\mathcal{X}))$; (b) labels for $\mathcal{T}(\mathcal{P}(\mathcal{X}))$; (c) matched code for $\mathcal{P}(\mathcal{X})$. 30

3.3 The encoder and decoder for the example in Figure 3.2. For the decoder, $-$ marks an event that can never occur, $\downarrow$ marks a case where the decoder knows it has not reached the end of $\gamma_X(x)$, and $\uparrow$ marks a case where the decoder would have decoded based on fewer symbol than are given. 32

3.4 Dividing the unit interval in (a) traditional arithmetic coding and (b) matched arithmetic coding for partition $\mathcal{P}(\mathcal{X})$ of Figure 3.2(a). (c) Matched arithmetic coding for sequence $a_7a_3a_4a_1a_2$. 38

3.5 Combining the groups in partition (a) to get partition (b); (c) the matched Huffman code rate for (a) is 2.25; (d) the matched Huffman code rate for (b) is 2.3. 44

3.6 Combining two groups ($\mathcal{G}_I$ and $\mathcal{G}_J$) into one group. 44

3.7 The partition pair $(\mathcal{P}(\mathcal{X}), \mathcal{P}(\mathcal{Y}))$ with $\mathcal{P}(\mathcal{X})$ shown in (a) and $\mathcal{P}(\mathcal{Y})$ shown in (b) gives a lossless, instantaneous MASC for the p.m.f. in Table 6.1(a). Replacing $\mathcal{P}(\mathcal{Y})$ with the partition shown in (c) fails to give a lossless, instantaneous MASC for the same p.m.f. 52

3.8 Three prefix/suffix relations between strings $\mathbf{c} \in \mathcal{S}_o$ and $\mathbf{s} \in \mathcal{S}_i$. 70

3.9 Definition of A, B, C, D, E, F, G. 78

3.10 Partition trees for the p.m.f. from Table 6.1(a) using (a) optimal arithmetic coding, (b) optimal Huffman coding, (c) arithmetic or Huffman coding with the approach in [1]. 81

3.11 Dimension-1 lossless and near-lossless MASC results for Tables 6.1(a) (top left), 6.1(b) (top right), 6.1(c) (middle left), 6.1(d) (middle right), 6.2(a) (bottom left), and 6.2(b) (bottom right). (H: Huffman code; A: arithmetic code; $S - W, P_e = 0$: Slepian-Wolf bound; $S - W, P_e = \epsilon$: bound for near-lossless MASC) 84

3.12 Performance of near lossless MASC as a function of coding dimension (Dim), for (a) $\alpha = \beta = 0.0002$ and (b) $\alpha = 0.002$, $\beta = 0.0002$. (H: Huffman code; A: arithmetic code) 85

3.13 Performance of four fast lossless SISC design algorithms. 89

3.14 Performance of four fast near-lossless SISC design algorithms. 92

3.15 Performance of ULSISC, $N = |\mathcal{X}| = |\mathcal{Y}|$. 95

4.1 Calculating Z_2 and Z_3 as deterministic functions of dither random variable Z_1. 102

4.2 MR-ECDQ implementation: (a) original image and dithered image; (b) rearranging pixels positions in multi-resolution coding. 108

4.3 The zigzag scanning pattern. 110

4.4 Various methods for improving the performance of ECDQ. 111

4.5 ECDQ+BWT's performance on a brain-scan image at different z_o values. . . 112

4.6 Comparison of the average performance of different ECDQ algorithms with SPIHT on a brain-scan image Brain1. 113

4.7 Comparison of the performance of ECDQ with SPIHT on image Lena. . . . 113

4.8 Comparison of the performance of ECDQ with SPIHT on image Airport. . . 114

4.9 Comparison of the performance of ECDQ with SPIHT on image Barbara. . . 114

4.10 Comparison of the performance of ECDQ with SPIHT on image Crowd. . . . 115

4.11 Comparing the rate achieved by MA-ECDQ with the minimal achievable rate at the same distortion for Gaussian sources. 118

6.1 Sample images from the GOES-8 weather satellites, used in BSVQ. 124

6.2 Examples of medical scan images used in ECDQ. 128

6.3 Test images used in ECDQ. 129

List of Tables

3.1 Counterexample used for proving Lemma 7. 68

3.2 An example where γ_X is not UD SISC as tested by TEST 1. In this table, $\mathcal{S}_i$ are classified according to y, hence $\mathcal{S}_i$ may contain repeated elements. This classification is convenient but not necessary. 71

3.3 Lossless SISC results for the p.m.f.s of Tables 6.1 and 6.2. In this table, $[H(X), R'_{SI,A}(X), R^{\star}_{SI,A}(X)]$ and $[R_H(X), R'_{SI,H}(X), R^{\star}_{SI,H}(X)]$ denote the optimal and Huffman results, respectively, for [traditional, SISC [1], optimal SISC] coding on X when Y is given as side information to the decoder. . . . 82

3.4 Comparing the running time of optimal design approaches. 86

3.5 Closeness to the optimal solution. $Ratio = R_{fast}/R_{opt}$, R_{fast} = average rate of the fast algorithm over K trials, $Prob.$ = probability of hitting the target rate. 90

3.6 Achievable rates for $N = 256$, complexity limit $C = N^2 = 65536$. 91

3.7 Good parameters 91

6.1 Sample p.m.f.s on alphabet $\mathcal{X} \times \mathcal{Y}$ with $\mathcal{X} = \mathcal{Y} = \{a_0, a_1, \ldots, a_6, a_7\}$. 125

6.2 Sample p.m.f.s $p(x, y)$ on alphabet $\mathcal{X} \times \mathcal{Y}$ with $\mathcal{X} = \mathcal{Y} = \{a_0, a_1, \ldots, a_{15}\}$. . . 126

6.3 An example $p(x, y)$ to show that $(\Gamma_1 \cap \Gamma_2) \cup (\Gamma_3 \cap \Gamma_4)$ be traditional UD code is not necessary. Note: * refers to $p(x, y) > 0$. 127

Chapter 1

Introduction

In the information age, network systems and applications have been growing rapidly. They provide an enormous variety of services, for example, Internet browsing, wireless communications, satellite communications, video conferencing, sensor networks, and distributed computing. However, further development of advanced network technologies is limited by the amount of information that can be sent through the corresponding networks. In networks where bandwidth is critically limited, it is imperative to use efficient data representations or source codes for optimizing network system performance.

I define a network to be any collection of nodes joined by communication links. In the most general case, every node in a network can communicate with every other node. Each node in a network has a collection of sources to be encoded into messages for transmission and a collection of sources for which it receives descriptions and builds reproductions. The traditional source coding strategy suggests that each node should compress each of its outgoing sources and decompress each of its incoming sources independently; thus this strategy is independent of the network topology.

In this thesis, I demonstrate the benefits of the alternative "network" approach in which network topology is incorporated into the compression system design. The data compression algorithms designed for use in network systems are called *network source codes*. Almost three decades have passed since the teams of Slepian and Wolf [2], Gray and Wyner [3], and Wyner and Ziv [4] demonstrated (for three simple networks) the enormous gains achievable in moving from source codes designed for traditional (single-sender, single-receiver) communication systems to network source codes. Yet the theory and design of network source codes remain largely unsolved problems. This thesis aims at tackling the problems of network source coding theory and design for some rudimentary network scenarios. In particular, I focus on code design for broadcast and multiple access systems.

Both lossless and lossy source codes are required for communications. Lossless codes are typically applied in applications like text transmission, banking, military data, and medical research, where perfect reconstruction of the data is required. Lossless codes are also used inside lossy codes. Lossy codes are generally applied in applications like video and image transmission, personal communications, Internet browsing, where some sacrifice in reconstruction fidelity is considered acceptable in light of the higher compression ratios of lossy codes. I work on both lossless and lossy network source codes in my thesis.

To tackle the problem of network source coding, I first choose two subsystems of a general network: broadcast system and multiple access networks, and treat each individually.

Chapter 2 treats source coding for broadcast systems. Broadcast systems represent applications where one node must simultaneously send either the same or different information to multiple nodes in the network. My research here gives results on both the theory and the practice of lossless and lossy broadcast system source codes, including derivation of the

theoretical limits of broadcast system source codes, a design algorithm for optimal broadcast system vector quantizers, a discussion of optimal code implementation and experimental results. This work also appears in [5, 6, 7].

Chapter 3 treats source coding for multiple access systems. Multiple access systems play a central role in an enormous variety of network technologies, where several transmitters send information to a single receiver. Multiple access source codes yield efficient data representations for multiple access systems when cooperation among the transmitters is not possible. In this chapter, I present the properties of instantaneous and uniquely decodable multiple access source codes and generalize the Huffman and arithmetic code design algorithms to attain the corresponding optimal instantaneous multiple access source codes for arbitrary joint source statistics. I further introduce a family of polynomial complexity code design algorithms that approximates the optimal solutions. Application to universal coding for multiple access networks when the joint source statistics are unknown *a priori* is also briefly discussed. Finally, I demonstrate algorithmic performance by showing experimental results on a variety of data sets. This work also appears in [8, 9, 10, 11, 12, 13, 14].

While most of the thesis treats lossless or nearly lossless coding, low complexity lossy source coding algorithms for network are also a topic of considerable importance and interest. Chapter 4 treats low complexity code design for a special type of broadcast source code called a multi-resolution source code. Multi-resolution source codes are compression algorithms for broadcast applications where all users desire the same information but different users may have different rate constraints.

Chapter 4 introduces and analyzes entropy constrained dithered quantizer for multi-resolution and multiple access source codes. For multi-resolution source coding, I demon-

strate that a scalar entropy constrained dithered quantizer can achieve performance at most a constant distance away from the n-dimensional optimum for arbitrary sources. For multiple access source coding, entropy constrained dithered quantization can also achieve performance close to the optimum for certain sources. My publications for this part include [15, 16, 17].

Finally Chapter 5 summarizes the contributions of this thesis and lists open problems for interested readers.

Chapter 2

Source Coding for Broadcast Systems

2.1 Introduction

A broadcast system is a network in which one transmitter simultaneously sends information to a collection of receivers. The information sent by the transmitter may include both common information, intended for a collection of receivers, and private information, intended for only one receiver. Broadcast systems play a role in many network communication environments. For example, a base station sending information to a collection of hand-held units in a wireless communication network is a broadcast system; similarly, a video conferencing system may be modeled as a collection of broadcast networks, where each node broadcasts one user's voice and video messages to all other nodes in the network. Paging systems and computer networks are also examples of network technologies where broadcast systems play a central role.

Broadcast system source codes (BSSC), introduced in [5, 6, 7], are source codes explicitly designed for broadcast systems. I first use a simplified broadcast system with two receivers

to illustrate the idea of BSSC. In this system, the transmitter sends private information X_1 to receiver 1, private information X_2 to receiver 2, and common information $X_{1,2}$ to both receivers. Receiver 1's reproduction of $(X_1, X_{1,2})$ is denoted by $\hat{X}_1 = (\hat{X}_{1,1}, \hat{X}_{1,2,1})$. Likewise receiver 2's reconstruction is $\hat{X}_2 = (\hat{X}_{2,2}, \hat{X}_{1,2,2})$.

To understand the potential benefit of BSSC, consider the following lossless coding example, illustrated by Figure 2.1. To achieve lossless source coding in a broadcast system using only traditional (single-encoder, single-decoder) source codes, we must encode the sources X_1, X_2, and $X_{1,2}$ independently using three separate encoders placed at the transmitter. That is, we independently map each source X_S ($S \in \{\{1\}, \{2\}, \{1,2\}\}$) into a channel codeword C_S from some rate-R_S lossless source code. Receiver 1 noiselessly receives C_1 and $C_{1,2}$, and independently decodes C_1 to give $\hat{X}_{1,1}$, and $C_{1,2}$ to give $\hat{X}_{1,2,1}$ using two separate decoders co-located at receiver 1. Likewise, receiver 2 receives C_2 and $C_{1,2}$, and uses independent decoders to reconstruct $\hat{X}_{2,2}$ and $\hat{X}_{1,2,2}$ in the same way. The decoder for the common information $C_{1,2}$ located at receiver 1 is the same as the decoder for $C_{1,2}$ at receiver 2. And thus $\hat{X}_{1,2,1} = \hat{X}_{1,2,2}$. Further, for this example all codes are lossless, and thus $\hat{X}_{1,2,1} = \hat{X}_{1,2,2} = X_{1,2}$, $\hat{X}_{1,1} = X_1$, and $\hat{X}_{2,2} = X_2$.

The above method decomposes the broadcast system into three separate encoder-decoder subsystems, each of which works on a single information source X_S independently. Thus, the rate R_S required by the decoder to losslessly reconstruct a random variable X_S must be greater than or equal to the entropy $H(X_S)$. As a result, the total rate $R_{*1} = R_1 + R_{1,2}$ received at receiver 1 and the total rate $R_{*2} = R_2 + R_{1,2}$ received at receiver 2 must be bounded as $R_{*1} \geq H(X_1) + H(X_{1,2})$, and $R_{*2} \geq H(X_2) + H(X_{1,2})$ to achieve lossless compression with the system described above.

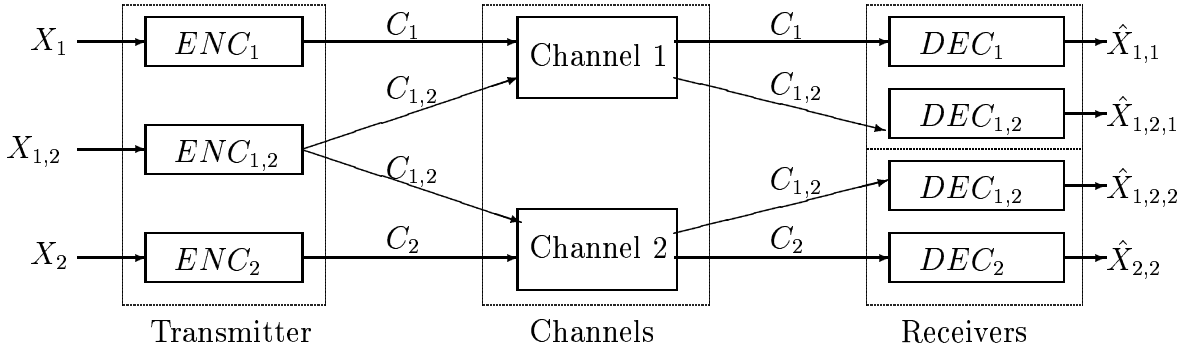


Figure 2.1: Traditional (single-encoder single-decoder) source codes for broadcast system.

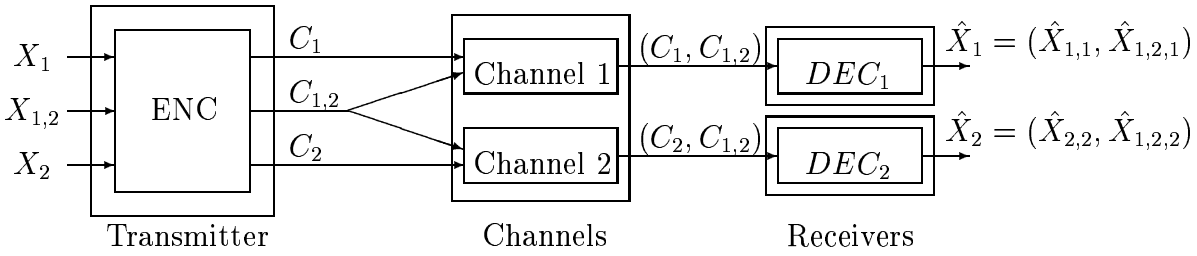


Figure 2.2: Broadcast System Source Coding.

Now consider replacing the collection of encoders at the transmitter with a single encoder at the transmitter and replacing the pair of decoders at each receiver by a single decoder at each receiver, as shown in Figure 2.2. I call the resulting code a *broadcast system source code*. Now, I jointly encode $(X_1, X_2, X_{1,2})$ into $(C_1, C_2, C_{1,2})$ with a single encoder. Receiver 1 noiselessly receives and jointly decodes $(C_1, C_{1,2})$, while receiver 2 noiselessly receives and jointly decodes $(C_2, C_{1,2})$. In this case, the encoding of X_1, X_2, and $X_{1,2}$ need not be independent, and likewise the decoding at each receiver is performed together. The following algorithm can thus be employed. Losslessly describe source $X_{1,2}$ with codeword $C_{1,2}$ from a rate-$H(X_{1,2})$ entropy code. Losslessly describe sources X_1 and X_2 with codewords C_1 and

C_2 from rate-$H(X_1|X_{1,2})$ and rate-$H(X_2|X_{1,2})$ conditional entropy codes conditioned on $X_{1,2}$ respectively. Receiver 1 receives both C_1 and $C_{1,2}$, while receiver 2 receives both C_2 and $C_{1,2}$. Each decoder can decode the common information (described by $C_{1,2}$) first and then use that common information to decode the received private information (described by C_1 or C_2). The total rate for receivers 1 and 2 respectively are $R_{*1} = H(X_{1,2}) + H(X_1|X_{1,2}) = H(X_1, X_{1,2})$ and $R_{*2} = H(X_{1,2}) + H(X_2|X_{1,2}) = H(X_2, X_{1,2})$, which are the minimal total rates possible for the given transmissions. When X_1 and $X_{1,2}$ are dependent random variables, this rate is smaller than the rate required by the previous method.

The above simple example demonstrates both the problems inherent in using traditional source codes in broadcast systems and the potential benefits of BSSC. In particular, the use of existing (single-encoder, single-decoder) source codes fails to take advantage of the specific characteristics of the broadcast system. Our most basic innovation in the proposal of BSSC is the assertion that source codes should be designed specially for broadcast systems. Dependent encoders at the transmitter and dependent decoders at each receiver are necessary in order to achieve optimal performance. Note, however, that the method given in the earlier example does not generalize to lossless codes with more than two receivers or to lossy source coding, where performance gains are likewise expected. Thus while the given method demonstrates the inefficiencies of using independent codes, it does not solve the general BSSC design problem. In later sections, I describe practical algorithms for optimal BSSC design.

Chapter Outline

This chapter treats the performance limits, optimal design, and empirical behavior of lossless and lossy BSSCs.

Section 2.2 defines notation. In Section 2.3, I derive the achievable rate region for lossless broadcast system source coding in an M-receiver system. Section 2.4 contains a detailed description of a locally optimal design algorithm for fixed- and variable-rate broadcast system vector quantization (BSVQ) for general M-receiver broadcast systems. The optimal design algorithm, which relies on a k-ary tree structure for some $k \geq 3$, generalizes the optimal multiple description vector quantizer design algorithm of [18]. In Section 2.5, I demonstrate the empirical performance of BSVQ for a 2-receiver broadcast system and compare that performance with the performance achievable with independent single-transmitter, single-receiver codes in the same system, thereby demonstrating the benefits of lossy BSSC. The key contributions of this chapter are summarized in Section 2.6.

2.2 Notation

Consider a single-transmitter, M-receiver broadcast system. I label the receivers by $1, \ldots, M$ and define $\mathcal{M} = \{1, \ldots, M\}$. Let $\mathcal{S}$ be the set of all nonempty subsets of $\mathcal{M}$. Then each set $S \in \mathcal{S}$ describes a group of receivers to whom a particular message may be sent. For an M-receiver system, the cardinality of $\mathcal{S}$ is $2^M - 1$, and thus there can be as many as $2^M - 1$ messages to be described. I assume an arbitrary but fixed ordering on $\mathcal{S}$; thus $\mathcal{S} = \{S(1), \ldots, S(2^M - 1)\}$. For any set $S \in \mathcal{S}$, $X_S[1], X_S[2], \ldots$ denotes the random source sequence to be described to all receivers in set S, and $\mathcal{X}_S$ denotes the corresponding source alphabet; thus for each $i \geq 1$, source symbol $X_S[i]$ may take on any value $x_S \in \mathcal{X}_S$. If $|S| > 1$, then X_S is the common information received by all of the receivers in S; if $|S| = 1$, then X_S is the private information received by the single receiver in S.

For any M-receiver broadcast system, a broadcast system source code contains one encoder and M decoders. In the most general case, the broadcast system transmitter uses the single encoder to jointly describe all sources X_S such that $S \in \mathcal{S}$. Each receiver $r \in \mathcal{M}$ uses a single decoder to jointly decode the descriptions of all sources X_S such that $r \in S$.

This work treats block BSSCs. I therefore begin by fixing blocklength or coding dimension n. For each $S \in \mathcal{S}$, I block the source sequence $\{X_S[i]\}_{i=1}^{\infty}$ into vectors of length n and use $X_S^n \in \mathcal{X}_S^n$ to denote a single n-vector. The encoder jointly encodes all source vectors X_S^n with $S \in \mathcal{S}$. For any receiver $r \in \mathcal{M}$, the decoder at receiver r jointly decodes all messages X_S^n with $S \in \mathcal{S}_r = \{S \in \mathcal{S} : r \in S\}$. Now for any set $\mathcal{W} \subseteq \mathcal{S}$, let $\mathbf{X}_{\mathcal{W}}^n$ denote the vector $(X_S^n : S \in \mathcal{W})$ and $\mathcal{X}_{\mathcal{W}}^n = \prod_{S \in \mathcal{W}} \mathcal{X}_S^n$ denote the corresponding product alphabet; thus $\mathbf{X}_{\mathcal{W}}^n$ can take on any value $\mathbf{x}_{\mathcal{W}}^n \in \mathcal{X}_{\mathcal{W}}^n$, where the components of $\mathbf{X}_{\mathcal{W}}^n$ and its alphabet $\mathcal{X}_{\mathcal{W}}^n$ are ordered according to the fixed ordering on $\mathcal{S}$. Then $\mathbf{X}_{\mathcal{S}}^n \in \mathcal{X}_{\mathcal{S}}^n$ is the vector of sources described by the encoder and $\mathbf{X}_{\mathcal{S}_r}^n \in \mathcal{X}_{\mathcal{S}_r}^n$ is the vector of sources intended for receiver r. For notational simplicity, I abbreviate this notation as $\mathbf{X}^n = \mathbf{X}_{\mathcal{S}}^n$, $\mathcal{X}^n = \mathcal{X}_{\mathcal{S}}^n$, $\mathbf{X}_r^n = \mathbf{X}_{\mathcal{S}_r}^n$ and $\mathcal{X}_r^n = \mathcal{X}_{\mathcal{S}_r}^n$. Finally, for each $S \in \mathcal{S}$ and $r \in S$, let $\hat{\mathcal{X}}_{S,r}$ denote the reproduction alphabet used by receiver r in its reproduction of source X_S. Then for any $S \in \mathcal{S}$, I use $\hat{X}_{S,r}^n$ to denote receiver r's reproduction of source vector X_S^n and $\hat{\mathbf{X}}_r^n = (\hat{X}_{S,r}^n : S \in \mathcal{S}_r)$ to denote the full vector of reproductions at receiver r. Thus $\hat{X}_{S,r}^n$ and $\hat{\mathbf{X}}_r^n$ can respectively take on any value $\hat{x}_{S,r}^n \in \hat{\mathcal{X}}_{S,r}^n$ and $\hat{\mathbf{x}}_r^n \in \hat{\mathcal{X}}_r^n = \prod_{S \in \mathcal{S}_r} \hat{\mathcal{X}}_{S,r}^n$.

The system encoder $\mathbf{a} : \mathcal{X}^n \to \mathcal{C}$ maps the space of input source vectors $\mathcal{X}^n$ to the transmitter's product channel codebook $\mathcal{C} = \prod_{S \in \mathcal{S}} \mathcal{C}_S$, where for each $S \in \mathcal{S}$, the individual channel codebook $\mathcal{C}_S$ is a subset of the set $\{0,1\}^*$ of finite binary strings. The encoder $\mathbf{a}$ comprises two parts, $\mathbf{a} = \gamma \circ \alpha$, where $\alpha : \mathcal{X}^n \to \mathcal{I}$, $\gamma : \mathcal{I} \to \mathcal{C}$, and $\circ$ denotes composition.

The intermediate alphabet $\mathcal{I}$ is the index set of the channel codebook $\mathcal{C}$, where index set $\mathcal{I}$ is a product alphabet of form $\mathcal{I} = \prod_{S \in \mathcal{S}} \mathcal{I}_S$ and $\mathcal{I}_S$ is the index set associated with source X_S. The first component α maps each source vector $\mathbf{X}^n \in \mathcal{X}^n$ to a channel codeword index $i \in \mathcal{I}$. In lossless codes, α is one-to-one and information lossless; while in lossy codes, α is many-to-one and information lossy. The function γ, which is information lossless, maps index i into the corresponding channel codeword $c_i \in \mathcal{C}$.

For each $r \in \mathcal{M}$, I use $\mathbf{t}_r : \mathcal{C} \to \mathcal{C}_r$ to denote the action of the channel seen by receiver r. Thus receiver r losslessly receives the subset of the channel codeword that is directly relevant to him. Receiver r's decoder $\mathbf{b}_r : \mathcal{C}_r \to \hat{\mathcal{X}}_r^n$ maps that receiver's product channel codebook $\mathcal{C}_r = \prod_{S \in \mathcal{S}_r} \mathcal{C}_S$ to reproduction space $\hat{\mathcal{X}}_r^n$ by way of index set $\mathcal{I}_r = \prod_{S \in \mathcal{S}_r} \mathcal{I}_S$. Thus $\mathbf{b}_r = \beta_r \circ \delta_r$, where $\delta_r : \mathcal{C}_r \to \mathcal{I}_r$ and $\beta_r : \mathcal{I}_r \to \hat{\mathcal{X}}_r^n$. Here $\mathcal{I}_r$ may be interpreted as the index set of the channel codebook $\mathcal{C}_r$. The function δ_r converts each received channel codeword $c_r \in \mathcal{C}_r$ to an index $i_r \in \mathcal{I}_r$, and the function β_r maps this index i_r to a reproduction vector $\hat{\mathbf{X}}_r^n \in \hat{\mathcal{X}}_r^n$.

For any $\mathbf{x}^n \in \mathcal{X}^n$, $\mathbf{a}(\mathbf{x}^n) = \gamma(\alpha(\mathbf{x}^n)) = \mathbf{c}$ for some $\mathbf{c} = (c_S : S \in \mathcal{S}) \in \mathcal{C}$, and for each receiver r, $\mathbf{c}_r = (c_S : S \in \mathcal{S}_r) \in \mathcal{C}_r$ and $\mathbf{b}_r(\mathbf{c}_r) = \beta_r(\delta_r(\mathbf{c}_r)) = \hat{\mathbf{x}}_r^n$ for some $\hat{\mathbf{x}}_r^n \in \hat{\mathcal{X}}_r^n$, where $c_S = \gamma_S(\alpha(\mathbf{x}^n))$. I use $\beta_{r,S}(\delta_r(\mathbf{c}_r)) = \hat{x}_{S,r}^n$ to denote receiver r's reproduction of x_S^n. In order for each decoder to be able to successfully decode a sequence of source vector descriptions, the product channel codebook $\mathcal{C}_r$ must be uniquely decodable for each $r \in \mathcal{M}$. I consider both fixed-and variable-rate channel codebooks.

2.3 Theoretical Bounds on Lossless Codes

In this section, I describe the achievable rate region for lossless source coding in an M-receiver broadcast system. The proof for this region borrows several ideas from the Slepian-Wolf theorem [19].

For our discussion of lossless BSSC performance bounds, I make the following assumptions. For each $S \in \mathcal{S}$, assume that $\mathcal{X}_S$ is a finite source alphabet. Further, assume that $\mathbf{X}[1], \mathbf{X}[2], \ldots$ is drawn i.i.d. with known probability mass function $p(\mathbf{x})$ on source alphabet $\mathcal{X}$. For any $\mathcal{U}, \mathcal{V}, \mathcal{W} \subseteq \mathcal{S}$, let $H(\mathbf{X}_\mathcal{W})$ denote the entropy of $\mathbf{X}_\mathcal{W}$ and $I(\mathbf{X}_\mathcal{U}; \mathbf{X}_\mathcal{V})$ the mutual information of $\mathbf{X}_\mathcal{U}$ and $\mathbf{X}_\mathcal{V}$.

Definition 1 *The set $A_{\epsilon,r}^{(n)} \subset \mathcal{X}_r^n$ of ϵ-typical n-vectors is defined by*

$$A_{\epsilon,r}^{(n)} = A_{\epsilon,r}^{(n)}(\mathbf{X}_r) = \{\mathbf{x}_r^n : |-(1/n)\log p(\mathbf{x}_\mathcal{W}^n) - H(\mathbf{X}_\mathcal{W})| < \epsilon, \quad \forall\ \mathcal{W} \subseteq \mathcal{S}_r\}.$$

For any $\mathcal{U}, \mathcal{V} \subseteq \mathcal{S}_r$, $A_{\epsilon,r}^{(n)}(\mathbf{X}_\mathcal{U}|\mathbf{x}_\mathcal{V}^n)$ is then defined as the set of $\mathbf{x}_\mathcal{U}^n$ vectors that are jointly ϵ-typical with a particular $\mathbf{x}_\mathcal{V}^n$ vector, i.e.,

$$A_{\epsilon,r}^{(n)}(\mathbf{X}_\mathcal{U}|\mathbf{x}_\mathcal{V}^n) = \{\mathbf{x}_\mathcal{U}^n : |-(1/n)\log p(\mathbf{x}_\mathcal{U}^n|\mathbf{x}_\mathcal{V}^n) - H(\mathbf{X}_\mathcal{U}|\mathbf{X}_\mathcal{V})| < \epsilon, \quad \forall\ \mathcal{U} \subseteq \mathcal{S}_r\}.$$

Definition 2 *A $((2^{nR_{S(1)}}, \ldots, 2^{nR_{S(|\mathcal{S}|)}}), n)$ reduced broadcast system source code for an M-receiver broadcast system consists of $|\mathcal{S}|$ encoder maps*

$$f_S : \mathcal{X}_S^n \to \{1, \ldots, 2^{nR_S}\}, S \in \mathcal{S}$$

and M decoder maps

$$g_r : \prod_{S \in \mathcal{S}_r} \{1, 2, \ldots, 2^{nR_S}\} \to \prod_{S \in \mathcal{S}_r} \mathcal{X}_S^n, \quad r \in \mathcal{M}.$$

The concept of a reduced BSSC is prompted by the following observation. Consider a 3-receiver system; the sources are $X_{\{1,2,3\}}$, $X_{\{1,2\}}$, $X_{\{1,3\}}$, $X_{\{2,3\}}$, $X_{\{1\}}$, $X_{\{2\}}$, $X_{\{3\}}$, abbreviated as X_{123}, X_{12}, X_{13}, X_{23}, X_1, X_2, and X_3 for simplicity. Much of the performance benefits associated with broadcast system source coding can be achieved through sequential description of the sources $\{X_S : S \in \mathcal{S}\}$. In particular, suppose that the encoder first describes X_{123}, then describes X_{12}, X_{13}, X_{23}, and finally describes X_1, X_2, X_3. For each receiver r, $r \in \{1, 2, 3\}$, the coding of X_{12}, X_{13}, X_{23} can be based on knowledge of X_{123}. That is, we can first losslessly describe X_{123}, then describe X_{12}, X_{13}, and X_{23} using conditional entropy codes conditioned on X_{123} and finally describe X_1, X_2 and X_3 using conditional entropy codes conditioned on the sources already decoded at each receiver. This sequential approach is decodable at all decoders since each decoder receives and decodes in advance all information needed by the conditional entropy codes' decoders. Note, however, that the coding of the common information which is shared by only 2 receivers, i.e., X_{12}, X_{13}, X_{23}, cannot be based on each other because, e.g., not all receivers that receive X_{12} also receive X_{13} or X_{23}. Given this observation, I seek codes in which the description of X_{12} is useful in the decoding of both X_{13} and X_{23} but not dependent on either. Only then can the description of X_{12} be successfully decoded at receivers 1 and 2. Toward this end, I replace the BSSC's single encoder with a collection of independent encoders and use an approach similar to the Slepian-Wolf coding strategy. The resulting code is the *reduced* BSSC. In the resulting reduced BSSC, the encoder at the transmitter encodes the sources independently, but the decoder at each receiver decodes the received sources jointly. The rate region of the lossless source code achieved by this coding scheme is given by Theorem 1.

Theorem 1 *The set of achievable rate vectors for reduced broadcast system source coding in a single-transmitter, M-receiver broadcast system is given by*

$$R_{\mathcal{W}} \geq \max_{r\in\mathcal{M}:\ \ \mathcal{W}\subseteq\mathcal{S}_r} H(\mathbf{X}_{\mathcal{W}}|\mathbf{X}_{\mathcal{S}_r\cap\mathcal{W}^c}), \quad \forall\ \mathcal{W}\subseteq\mathcal{S}_r \text{ for some } \ r\in\mathcal{M}, \tag{2.1}$$

where $R_{\mathcal{W}} = \sum_{S\in\mathcal{W}} R_S$ and superscript c denotes the set complement operator.

Proof of Achievability

The proof follows an approach similar to the proof of the Slepian-Wolf theorem.

I first randomly design a collection of encoders as follows. For each $S \in \mathcal{S}$, independently assign every $x_S^n \in \mathcal{X}_S^n$ to one of 2^{nR_S} bins according to a uniform distribution on $\{1, 2, \ldots, 2^{nR_S}\}$. The resulting mappings are the encoders $\{f_S\}$. Reveal f_S to all receivers r, such that $S \in \mathcal{S}_r$.

The decoders are then designed as follows. For any $r \in \mathcal{M}$, receiver r receives the index vector $(i_S, S \in \mathcal{S}_r)$ and declares $\hat{\mathbf{x}}_r^n = \mathbf{x}_r^n$, if there is one and only one n-vector $\mathbf{x}_r^n \in \mathcal{X}_r^n$ such that for all $S \in \mathcal{S}_r$, $f_S(x_S^n) = i_S$ and $\mathbf{x}_r^n \in A_{\epsilon,r}^{(n)}$. Otherwise the decoder declares an error.

For any $r \in \mathcal{M}$, receiver r's probability of error is defined as $P_{e,r}^{(n)} = P(\hat{\mathbf{X}}_r \neq \mathbf{X}_r)$. We have an error if $\mathbf{X}_r^n \notin A_{\epsilon,r}^{(n)}$ or if there is another typical sequence in the same bin. Let $\mathbf{X}_r$ be drawn according to $p(\mathbf{x}_r)$, $\bar{x}_S^n \in \mathcal{X}_S^n$, $\bar{\mathbf{x}}_{\mathcal{W}}^n \in \mathcal{X}_{\mathcal{W}}^n$. Define the events

$$\begin{aligned} E_{0,r} &= \{\ \mathbf{X}_r^n \notin A_{\epsilon,r}^{(n)}\ \}, \\ E_{\mathcal{W},r} &= \{\ \exists \bar{\mathbf{x}}_{\mathcal{W}}^n : \forall S\in\mathcal{W},\ \bar{x}_S^n \neq X_S^n,\ f_S(\bar{x}_S^n) = f_S(X_S^n) \\ &\qquad \text{and } (\bar{\mathbf{x}}_{\mathcal{W}}^n, \mathbf{X}_{\mathcal{S}_r\cap\mathcal{W}^c}^n) \in A_{\epsilon,r}^{(n)}\ \} \quad \forall\ \mathcal{W}\subseteq\mathcal{S}_r. \end{aligned}$$

Hence by the union bound,

$$P_{e,r}^{(n)} = P((\cup_{\mathcal{W}\subseteq\mathcal{S}_r} E_{\mathcal{W},r}) \cup E_{0,r}) \leq P(E_{0,r}) + \sum_{\mathcal{W}\subseteq\mathcal{S}_r} P(E_{\mathcal{W},r}).$$

First consider $P(E_{0,r})$. By the AEP, $P(E_{0,r}) \to 0$ and therefore for any $\epsilon > 0$, $P(E_{0,r}) < \epsilon$ for n large enough. For any $\mathcal{W} \subseteq \mathcal{S}_r$, bound $P(E_{\mathcal{W},r})$ as

$$\begin{aligned}
& P(E_{\mathcal{W},r}) \\
&= P\{\exists \bar{\mathbf{x}}_{\mathcal{W}}^n : \forall S \in \mathcal{W}, \bar{x}_S^n \neq X_S^n, f_S(\bar{x}_S^n) = f_S(X_S^n) \text{ and } (\bar{\mathbf{x}}_{\mathcal{W}}^n, \mathbf{X}_{\mathcal{S}_r\cap\mathcal{W}^c}^n) \in A_{\epsilon,r}^{(n)}\} \\
&= \sum_{\mathbf{x}_r^n} p(\mathbf{x}_r^n) P\{\exists \bar{\mathbf{x}}_{\mathcal{W}}^n : \forall S \in \mathcal{W}, \bar{x}_S^n \neq x_S^n, f_S(\bar{x}_S^n) = f_S(x_S^n), (\bar{\mathbf{x}}_{\mathcal{W}}^n, \mathbf{x}_{\mathcal{S}_r\cap\mathcal{W}^c}^n) \in A_{\epsilon,r}^{(n)}\} \\
&\leq \sum_{\mathbf{x}_r^n} p(\mathbf{x}_r^n) (\prod_{S\in\mathcal{W}} 2^{-nR_S}) |A_{\epsilon,r}^{(n)}(\mathbf{X}_{\mathcal{W}} | \mathbf{x}_{\mathcal{S}_r\cap\mathcal{W}^c}^n)| \\
&\leq (\prod_{S\in\mathcal{W}} 2^{-nR_S}) 2^{n(H(\mathbf{X}_{\mathcal{W}}|\mathbf{X}_{\mathcal{S}_r\cap\mathcal{W}^c})+\epsilon)}.
\end{aligned}$$

which goes to 0 if $\sum_{S\in\mathcal{W}} R_S > H(\mathbf{X}_{\mathcal{W}}|\mathbf{X}_{\mathcal{S}_r\cap\mathcal{W}^c})$. Hence for sufficiently large n, $P(E_{\mathcal{W},r}) < \epsilon$.

Combining the above results for all $r \in \mathcal{M}$ and $\mathcal{W} \subseteq \mathcal{S}_r$, I can bound the probability of error $P_e^{(n)}$ of the whole system for n large enough. In particular, if

$$\sum_{S\in\mathcal{W}} R_S > H(\mathbf{X}_{\mathcal{W}}|\mathbf{X}_{\mathcal{S}_r\cap\mathcal{W}^c}), \ \forall r \in \mathcal{M}, \forall \mathcal{W} \subseteq \mathcal{S}_r \tag{2.2}$$

then

$$P_e^{(n)} = \sum_{r\in\mathcal{M}} P_{e,r}^{(n)} \leq \sum_r (P(E_{0,r}) + \sum_{\mathcal{W}\subseteq\mathcal{S}_r} P(E_{\mathcal{W},r})) < \sum_r 2^{|\mathcal{S}_r|}\epsilon = K\epsilon,$$

where K is a constant. The conditions in (2.2) define the rate region of (2.1). Since the expected error probability for a random code is small, there exists at least one code $(\{f_S^\star\}_S, \{g_r^\star\}_r)$ with probability of error $< K\epsilon$, and we can construct a sequence of codes with $P_e^{(n)} \to 0$. □

Proof of Converse

Consider a reduced BSSC with fixed encoders $\{f_S\}$, decoders $\{g_r\}$ and probability of error $P_e^{(n)} \to 0$ as $n \to \infty$. Let $I_S = f_S(X_S^n)$. Since $P_e^{(n)} = \sum_{r \in \mathcal{M}} P_{e,r}^{(n)}$ goes to 0 when $n \to \infty$, we have for all r, $P_{e,r}^{(n)} \to 0$, as $n \to \infty$. For any $\mathcal{W} \subseteq \mathcal{S}$, let $\mathbf{I}_\mathcal{W}$ denote the index vector $(I_S : S \in \mathcal{W})$, and abbreviate the notation $\mathbf{I}_{\mathcal{S}_r}$ as $\mathbf{I}_r$. Then by Fano's inequality

$$H(\mathbf{X}_r^n|\mathbf{I}_r) \leq P_{e,r}^{(n)} n \sum_{S \in \mathcal{S}_r} (\log |\mathcal{X}_S|) + 1 = n\epsilon_n \text{ and } \; H(\mathbf{X}_\mathcal{W}^n|\mathbf{X}_{\mathcal{S}_r \cap \mathcal{W}^c}^n, \mathbf{I}_r) \leq n\epsilon_n \;\; \forall \mathcal{W} \subseteq \mathcal{S}_r,$$

where $\epsilon_n \to 0$ as $n \to \infty$. Then any $r \in \mathcal{M}$, and any $\mathcal{W} \subseteq \mathcal{S}_r$,

$$\begin{aligned}
n \sum_{S \in \mathcal{W}} R_S &\geq H(\, \mathbf{I}_\mathcal{W} \,) \\
&\geq H(\, \mathbf{I}_\mathcal{W} \mid \mathbf{X}_{\mathcal{S}_r \cap \mathcal{W}^c}^n \,) \\
&= I(\, \mathbf{X}_\mathcal{W}^n;\, \mathbf{I}_\mathcal{W} \mid \mathbf{X}_{\mathcal{S}_r \cap \mathcal{W}^c}^n \,) + H(\, \mathbf{I}_\mathcal{W} \mid \mathbf{X}_\mathcal{W}^n,\, \mathbf{X}_{\mathcal{S}_r \cap \mathcal{W}^c}^n \,) \\
&= I(\, \mathbf{X}_\mathcal{W}^n;\, \mathbf{I}_\mathcal{W} \mid \mathbf{X}_{\mathcal{S}_r \cap \mathcal{W}^c}^n \,) + H(\, \mathbf{I}_\mathcal{W} \mid \mathbf{X}_r^n \,) \\
&= I(\, \mathbf{X}_\mathcal{W}^n;\, \mathbf{I}_\mathcal{W} \mid \mathbf{X}_{\mathcal{S}_r \cap \mathcal{W}^c}^n \,) \\
&= H(\, \mathbf{X}_\mathcal{W}^n \mid \mathbf{X}_{\mathcal{S}_r \cap \mathcal{W}^c}^n \,) - H(\, \mathbf{X}_\mathcal{W}^n \mid \mathbf{X}_{\mathcal{S}_r \cap \mathcal{W}^c}^n,\, \mathbf{I}_\mathcal{W} \,) \\
&= H(\, \mathbf{X}_\mathcal{W}^n \mid \mathbf{X}_{\mathcal{S}_r \cap \mathcal{W}^c}^n \,) - H(\, \mathbf{X}_\mathcal{W}^n \mid \mathbf{X}_{\mathcal{S}_r \cap \mathcal{W}^c}^n,\, \mathbf{I}_\mathcal{W},\, \mathbf{I}_{\mathcal{S}_r \cap \mathcal{W}^c} \,) \\
&= H(\, \mathbf{X}_\mathcal{W}^n \mid \mathbf{X}_{\mathcal{S}_r \cap \mathcal{W}^c}^n \,) - H(\, \mathbf{X}_\mathcal{W}^n \mid \mathbf{X}_{\mathcal{S}_r \cap \mathcal{W}^c}^n,\, \mathbf{I}_r \,) \\
&\geq H(\, \mathbf{X}_\mathcal{W}^n \mid \mathbf{X}_{\mathcal{S}_r \cap \mathcal{W}^c}^n \,) - n\epsilon_n \\
&= nH(\, \mathbf{X}_\mathcal{W} \mid \mathbf{X}_{\mathcal{S}_r \cap \mathcal{W}^c} \,) - n\epsilon_n
\end{aligned}$$

Dividing by n and taking the limit as $n \to \infty$, we have the desired converse. □

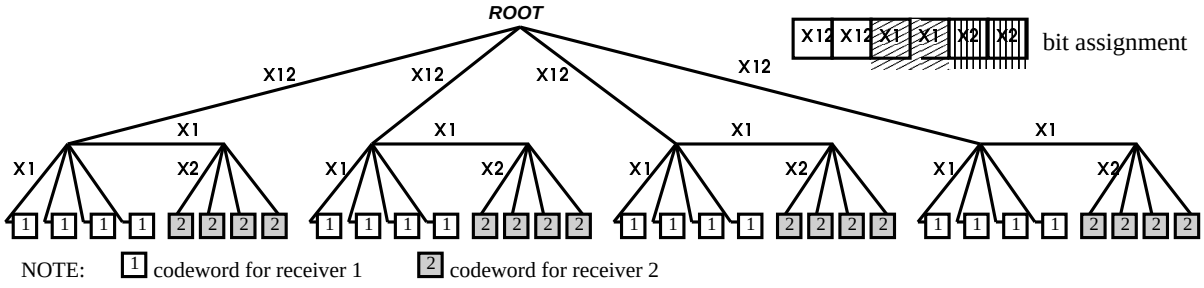


Figure 2.3: The k-ary tree structure for 2-receiver broadcast system lossy code.

2.4 Lossy Code Design

Our broadcast system vector quantization (BSVQ) design algorithm is based on a tree structure similar to that used for the multiple description vector quantization algorithm [18]. To understand that tree structure, first consider the k-ary tree with $k = 5$ shown in Figure 2.3. The given tree is intended for use in an n-dimensional, 2-receiver BSVQ with index set $\mathcal{I}_{12} = \mathcal{I}_1 = \mathcal{I}_2 = \mathcal{I} = \{1, 2, 3, 4\}$. Each horizontal branch of the tree denotes an unreceived source description, while each downward path denotes a received index from $\mathcal{I}$. For the code given in Figure 2.3, each receiver receives 2 indices: receiver 1 receives $\mathbf{i}_1 = (i_{12}, i_1)$ and receiver 2 gets $\mathbf{i}_2 = (i_{12}, i_2)$. Thus each receiver can receive one of 16 possible messages. Associated with each possible message is a vector $(\hat{X}_S^n : S \in \mathcal{S})$ of reproduction vectors or *codewords*. The codewords reside at leaves of the tree. In particular, the codewords for receivers 1 and 2 are indicated by $\boxed{1}$ and $\boxed{2}$, respectively, in Figure 2.3. The given figure shows the indices in order i_{12}, i_1, i_2 from top to bottom in the tree. (The order is arbitrary.) Since receiver 1 receives i_{12} and i_1 but not i_2, the codewords used by receiver 1 sit at the 16 nodes corresponding to two downward paths followed by a horizontal path through the tree. For receiver 2's codewords the horizontal path is the second step rather than the third.

The given structure generalizes to arbitrary M-receiver broadcast systems (by adding more levels to the tree) with arbitrary finite index sets (by allowing an arbitrary number of downward branches) and may be used for both fixed-rate coding (by describing each index using its natural binary description) and variable-rate coding (by describing each index using an appropriate entropy code).

I now adapt the generalized Lloyd algorithm [20, 21] to BSSC design. For any receiver r and any set $S \in \mathcal{S}_r$, let $D_{S,r} = (1/n)\mathrm{E}_S[d(X_S^n, \hat{X}_{S,r}^n)]$ denote the per symbol expected distortion of the reproduction of source X_S made by receiver r. For any set $S \in \mathcal{S}$, let $R_S = (1/n)\mathrm{E}_S[R_S(X_S^n)]$ denote the average rate used to transmit X_S, where $R_S(X_S^n)$ denotes the rate used to describe source vector X_S^n.

Let $\mathbf{D} = (D_{S,r} : S \in \mathcal{S}_r, r \in \mathcal{M})$ and $\mathbf{R} = (R_S : S \in \mathcal{S})$. The set of distortion-rate pairs $(\mathbf{D}, \mathbf{R})$ achievable through BSVQ (of arbitrary dimension) defines a convex region in the $(\sum_r |\mathcal{S}_r| + |\mathcal{S}|)$-dimensional distortion-rate space. This convex hull is entirely characterized by its support functional $J = \sum_r \sum_{S \in \mathcal{S}_r} \mu_{S,r}(D_{S,r} + \lambda_S R_S)$, where $\mu_{S,r}$ and λ_S are the associated Lagrangian constants. The functional J may be viewed as a Lagrangian for minimizing any combination of rates and distortions subject to constraints on the remaining quantities.

The goal of the algorithm is to minimize

$$\begin{aligned} & J(\alpha, \{\gamma_S\}, \{\delta_r\}, \{\beta_{r,S}\}) \\ = \; & \frac{1}{n}\mathrm{E}_{X_S^n}[\sum_{S \in \mathcal{S}_r, r} \mu_{S,r}[d(X_S^n, \beta_{r,S}(\delta_r(\mathbf{t}_r(\gamma(\alpha(\mathbf{X}^n)))))) + \lambda_S |\gamma_S(\alpha(X_S^n))|]], \end{aligned}$$

where α, $\{\gamma_S\}$, $\{\delta_r\}$ and $\{\beta_{r,S}\}$ are the encoders and decoders defined in Section 2.2. The design of the BSVQ employs a convergent iterative descent approach reminiscent of the

generalized Lloyd algorithm. Each iteration proceeds as follows.

1. **Optimize the encoder** α. For a given $\{\gamma_S\}$, $\{\delta_r\}$ and $\{\beta_{r,S}\}$, the optimal encoder $\alpha^\star$ is the one that maps each source vector $\mathbf{x}^n$ to the index i, such that the reproductions $\hat{x}^n_{S,r}$ will result in the smallest J. Thus $\alpha^\star$ is given by

$$\alpha^\star(x^n) = \arg\min_{i\in\mathcal{I}} \sum_{S\in\mathcal{S}_r,r} \mu_{S,r}[d(X^n_S, \beta_{r,S}(\delta_r(\mathbf{t}_r(\gamma(i))))) + \lambda_S|\gamma_S(i)|].$$

2. **Optimize the decoder** β_r. For a given α, γ and $\{\delta_r\}_{r\in\mathcal{M}}$, the optimal decoder $\beta^\star_r(i_r)$, $r \in \mathcal{M}$, is the one that minimizes the expected reproduction distortion given i_r. The mathematical expression for $\beta^\star_r(i_r)$ is given by

$$\beta^\star_r(i_r) = \arg\min_{\hat{\mathbf{x}}^n_r\in\hat{\mathcal{X}}^n_r} \sum_S \mu_{S,r}\left[\mathrm{E}_{X^n_S}\left[d(X^n_S, \hat{x}^n_{S,r})|\delta_r(\mathbf{t}_r(\gamma(\alpha(\mathbf{X}^n)))) = i_r\right]\right].$$

   In particular, for the squared error distortion measure, $\beta^\star_r(i_r)$ is the expected value of all the source vectors $\mathbf{X}^n$ such that $\delta_r(\mathbf{t}_r(\gamma(\alpha(\mathbf{X}^n)))) = i_r$, and is given by

$$\beta^\star_r(i_r) = \hat{x}^n_{S,r} = \mathrm{E}_{X^n_S}[X^n_S|\delta_r(\mathbf{t}_r(\gamma(\alpha(\mathbf{X}^n)))) = i_r].$$

3. **Optimize the lossless encoder** γ **and the lossless decoder** δ_r. For a given α and $\{\beta_r\}_{r\in\mathcal{M}}$, the optimal decoder γ and $\{\delta_r\}_{r\in\mathcal{M}}$ are the ones that minimize the expected description length.

2.5 Experimental results

I include results for experiments performed for the two-receiver broadcast system using the algorithm described in Section 2.4. The experiments are performed on the satellite weather

data sets, obtained courtesy of NASA and the University of Hawaii. It contains images from weather satellite GOES-8, which records 8-bit greyscale images in three frequency bands (visible, infrared 1, and infrared 2). Each image is cropped to 512×512 pixels. Some sample images are shown in the Appendix.

The training set contains three sets of satellite weather images. Each set contains eight images from the same frequency band. The visible frequency band set is used as training data for $X_{1,2}$; the infrared 1 and infrared 2 sets are used for X_1 and X_2, respectively. A non-overlapping test set contains three image sets with four images each and is assigned to the three sources similarly.

To choose the Lagrangian constants, I make several assumptions: (1) no receiver cares about the private information of any other, thus $\mu_{\{2\},1} = \mu_{\{1\},2} = 0$; (2) there is a trade-off between the two private informations, thus $\mu_{\{1\},1} = \sigma$, $\mu_{\{2\},2} = 1 - \sigma$, and (3) the common information has equal importance for both receivers, thus $\mu_{\{1,2\},1} = \mu_{\{1,2\},2} = \theta/2$.

Since the 9-dimensional rate-distortion space associated with this problem is difficult to visualize, the rates are fixed as follows. Each receiver is assigned a fixed rate of 4 bits per pixel (bpp), and the system uses a total rate of 6 bpp. To meet these rate constraints, in fixed-rate coding, I use a 5-ary tree and set $\lambda_S = 0$. In variable-rate coding, I use a 17-ary tree and for each (θ, σ) pair adjust λ_S to meet the desired rate constraint. By fixing the rates in this way, I can focus on the convex hull of the achieved distortions $\mathbf{D} = ((\mathbf{D}_{\{1,2\},1} + \mathbf{D}_{\{1,2\},2})/2, \mathbf{D}_{\{1\},1}, \mathbf{D}_{\{2\},2})$ in a 3-dimensional space.

For comparison purposes, I use either fixed-rate VQ or entropy constrained VQ (ECVQ) to independently encode and decode the sources, and I compare the resulting achievable distortions $\mathbf{D} = (\mathbf{D}_{\{1,2\}}, \mathbf{D}_{\{1\}}, \mathbf{D}_{\{2\}})$ with those of BSVQ. Since the rate assigned to each

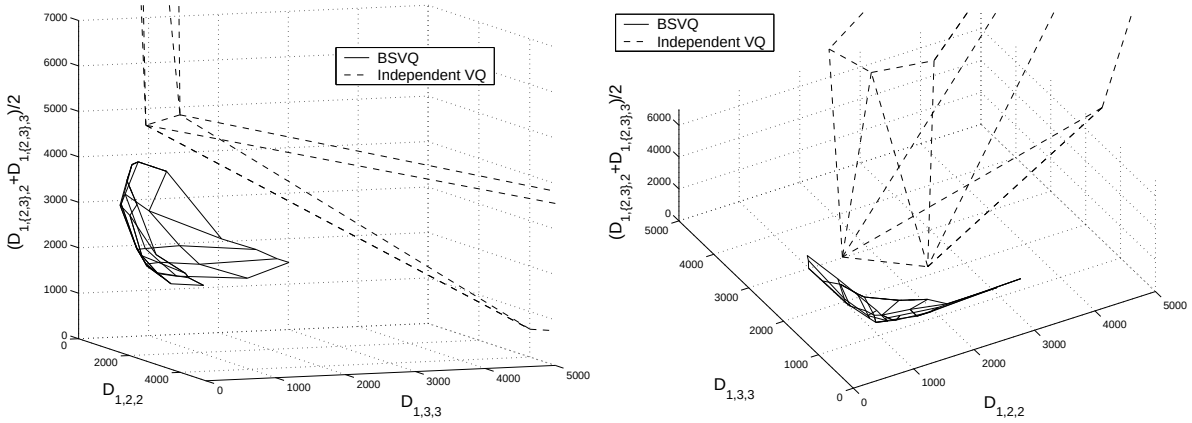


Figure 2.4: Fixed-rate BSVQ vs. fixed-rate VQ.

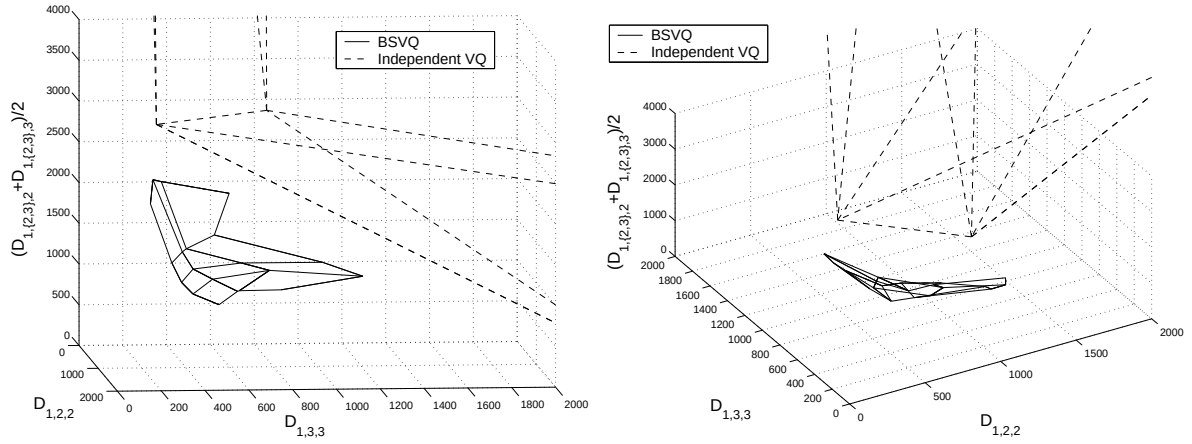


Figure 2.5: Variable-rate BSVQ vs. ECVQ.

receiver is identical for the BSVQ and for the independent codes, the lower the convex hull of distortions in 3-dimensional space, the better the performance.

Figure 2.4 compares the distortion triple of fixed-rate BSVQ with that of fixed-rate independent VQ, showing the 3-dimensional distortion space from two different angles. Figure 2.5 compares the distortion triple of variable-rate BSVQ with that of independent ECVQ. In both comparisons, the convex hull achieved by the BSVQ (solid line) is significantly lower than that achieved by the independent source code (dashed line) in the 3-dimensional space.

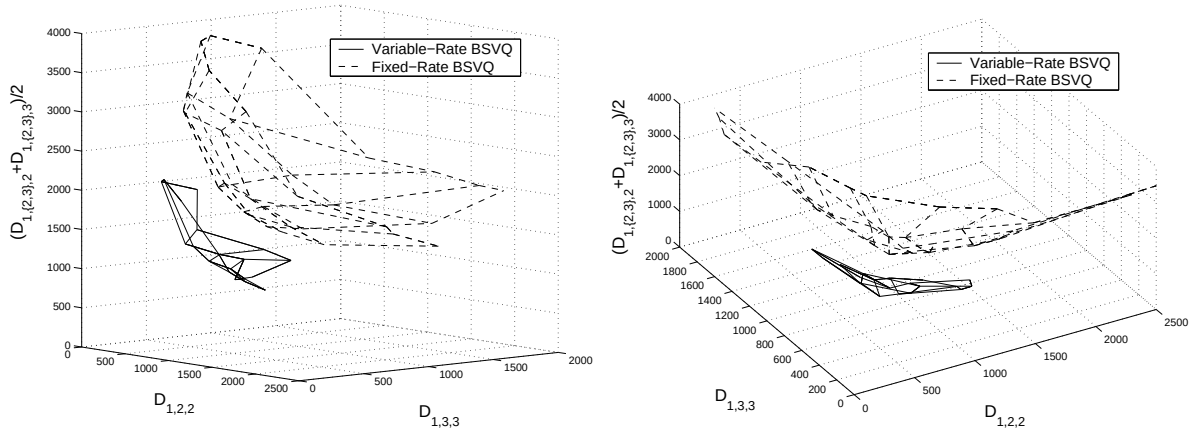


Figure 2.6: Variable-rate BSVQ vs. fixed-rate BSVQ.

Thus it demonstrates the performance benefits of BSVQs over independent source codes in a 2-receiver system.

Finally, Figure 2.6 compares fixed-rate BSVQ with variable-rate BSVQ. As with independent codes, variable-rate codes (of a given dimension) outperform fixed-rate codes (of the same dimension).

2.6 Summary

In this chapter, I prove the theoretical limits of lossless source coding performance in M-receiver broadcast systems. This proof also shows that only through joint coding can we hope to achieve optimal performance for broadcast systems. A detailed description of the practical algorithm for BSVQ is given, followed by experimental results, which show the benefits of BSVQ over independent coding.

Chapter 3

Source Coding for Multiple Access Systems

3.1 Introduction

A multiple access network comprises multiple transmitters sending information to a single receiver. One example of a multiple access system is a sensor network, where separately located sensors send correlated information to a central processing unit.

Multiple access source codes (MASCs) (also known as Slepian-Wolf or distributed source codes) yield efficient data representations for multiple access systems when cooperation among the transmitters is not possible. In the MASC configuration shown in Figure 3.1(a), two encoders independently describe information to a single decoder. The decoder uses the received pair of descriptions to reconstruct the original data sequences. In [19], Slepian and Wolf describe all rate pairs achievable with coding dimension $n \to \infty$ and probability of decoding error $P_e^{(n)} \to 0$. (See Figure 3.1(b).)

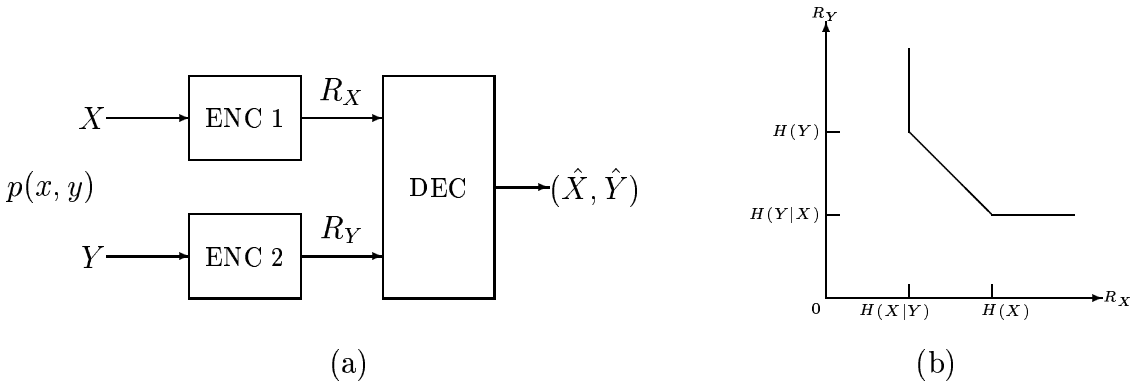


Figure 3.1: (a) An MASC and (b) the Slepian-Wolf achievable rate region for MASCs.

This chapter treats MASCs ($n < \infty$) for both the lossless ($P_e^{(n)} = 0$) and "near-lossless" ($P_e^{(n)} \to 0$) cases. The discontinuity in the limiting rate region at $P_e^{(n)} = 0$ [22][1] motivates the interest in near-lossless coding. For finite n, this discontinuity occurs at $P_e^{(n)} \geq \min\{p^n(x^n, y^n) : p^n(x^n, y^n) > 0\}$ rather than $P_e^{(n)} = 0$. Given their superior rate capabilities, near-lossless codes are useful where small error probabilities are acceptable (e.g., as entropy codes in lossy MASCs).

Prior work on MASCs for $n < \infty$ focuses primarily on the special case of a lossless instantaneous *side information source code* (SISC), where the decoder knows Y and the goal is to uniquely describe X using the smallest possible average rate. Work on lossless instantaneous SISC design appears in [22, 1]; these algorithms are suboptimal by [23]. Work on properties of optimal SISCs includes [24], which uses a graph-theoretic framework to derive bounds on the minimal expected rate in terms of the graph entropy, and [23], which describes necessary and sufficient conditions for the existence of a code with a given set of

[1]For example, if $p(x,y) > 0$ for all $(x,y) \in \mathcal{X} \times \mathcal{Y}$, then achieving $P_e^{(n)} = 0$ requires $R_X + R_Y \geq H(X) + H(Y)$ for all n. In contrast, $R_X + R_Y \geq H(X,Y)$ is achievable when $n \to \infty$ and $P_e^{(n)} \to 0$.

codeword lengths when the alphabet size of Y is two [23]. A design algorithm for lossless SISCs and MASCs for sources X and Y guaranteed to meet a maximal Hamming distance constraint appears in [25, 26].

Near-lossless codes are a special case of lossy codes with a Hamming distortion measure and an asymptotically negligible distortion. Work on lossy MASCs appears in [27, 28, 25, 26, 7, 29, 30, 31].

Chapter Outline

This chapter derives properties of MASCs and uses these properties to extend the definitions of Huffman and arithmetic codes to achieve corresponding lossless and near-lossless MASCs, giving the first constructive algorithm for building optimal lossless and near-lossless instantaneous SISCs and MASCs for general sources. The definitions and methods apply to arbitrary discrete-alphabet sources. While the encoding and decoding complexities of the proposed optimal MASCs are comparable to the corresponding complexities for traditional (single-sender, single-receiver) Huffman and arithmetic codes, the design complexities for the optimal MASCs are high. Since high design complexities seem to be unavoidable (in [32], Koulgi et al. show that even the lossless SISC design problem is NP-hard), I also consider low complexity approximate solutions.

The remainder of this chapter is organized as follows. Section 3.2 contains generalizations of the Huffman and arithmetic code design algorithms to the lossless instantaneous SISC problem. Section 3.3 extends these results to general MASCs. Section 3.4 treats the near-lossless MASC problem. Section 3.5 gives a family of low complexity sub-optimal design algorithms. Section 3.6 analyzes properties of lossless uniquely decodable MASCs.

Section 3.7 contains experimental results for both optimal and sub-optimal coding algorithms and an application of these coding algorithms to universal coding when the source distribution is unknown *a priori*. Section 3.8 gives a summary of the key contributions of this chapter.

3.2 Lossless Instantaneous Side Information Source Codes

3.2.1 Problem Statement

Let X and Y be memoryless sources with joint probability mass function (p.m.f.) $p(x, y)$ on finite alphabet $\mathcal{X} \times \mathcal{Y}$. I use $p_X(x)$ and $p_Y(y)$ to denote the marginals of $p(x, y)$ with respect to X and Y, dropping the subscripts when they are clear from the argument to give $p(x) = p_X(x)$ and $p(y) = p_Y(y)$. A *lossless instantaneous MASC* for joint source (X, Y) consists of two encoders $\gamma_X : \mathcal{X} \to \{0,1\}^\star$ and $\gamma_Y : \mathcal{Y} \to \{0,1\}^\star$ and a decoder $\gamma^{-1} : \{0,1\}^\star \times \{0,1\}^\star \to \mathcal{X} \times \mathcal{Y}$. Here $\gamma_X(x)$ and $\gamma_Y(y)$ are the binary descriptions of x and y, and the probability of decoding error is $P_e = \Pr(\gamma^{-1}(\gamma_X(X), \gamma_Y(Y)) \neq (X, Y))$. This section treats lossless coding, where $P_e \equiv 0$. Further, I concentrate exclusively on instantaneous codes, where for any input sequences $x_1, x_2, x_3, \ldots$ and $y_1, y_2, y_3, \ldots$ with $p(x_1, y_1) > 0$ the decoder reconstructs (x_1, y_1) by reading only the first $|\gamma_X(x_1)|$ bits from $\gamma_X(x_1)\gamma_X(x_2)\gamma_X(x_3)\ldots$ and the first $|\gamma_Y(y_1)|$ bits from $\gamma_Y(y_1)\gamma_Y(y_2)\gamma_Y(y_3)\ldots$ (without prior knowledge of these lengths).

When Y is perfectly known to the decoder, the problem reduces to the SISC problem. This scenario describes MASCs where γ_Y encodes Y using a traditional code for p.m.f. $\{p(y)\}_{y \in \mathcal{Y}}$ so that γ_X can encode X assuming that the decoder knows Y. In this case,

$\gamma^{-1} : \{0,1\}^\star \times \mathcal{Y} \to \mathcal{X}$. If the decoder can correctly reconstruct x_1 by reading only the first $|\gamma_X(x_1)|$ bits of $\gamma_X(x_1)\gamma_X(x_2)\gamma_X(x_3)\ldots$, then (γ_X, γ^{-1}) is a *lossless instantaneous SISC*. I wish to design a lossless instantaneous SISC that achieves the lowest possible expected rate. I treat code design for general MASCs in Section 3.3.

Lemma 1 *(SISC Prefix Property): Code γ_X is a lossless instantaneous SISC for X given Y if and only if for each x, x', y with $p(x,y) > 0$ and $p(x',y) > 0$, $\{\gamma_X(x), \gamma_X(x')\}$ is prefix-free.*

Proof: Necessary: If there exists some $y \in \mathcal{Y}$ and $x, x' \in \mathcal{X}$ for which $p(x,y) > 0$, $p(x',y) > 0$, and $\gamma_X(x)$ is a prefix of $\gamma_X(x')$, then the codewords for x and x' cannot be instantaneously distinguished when $Y = y$. Sufficient: The decoder receives Y and performs the mapping $\gamma^{-1}(\cdot, Y) : \{0,1\}^\star \to \{x \in \mathcal{X} : p(x,Y) > 0\}$. Since $\{\gamma_X(x) : p(x,Y) > 0\}$ is prefix-free, the code is instantaneous. □

Distinct symbols $x, x' \in \mathcal{X}$ are *confusable* under $p(x,y)$, written $x \doteqdot x'$, if $p(x,y) > 0$ and $p(x',y) > 0$ for some $y \in \mathcal{Y}$. By Lemma 1, an SISC's descriptions of x and x' can be identical ($\gamma_X(x) = \gamma_X(x')$) or the description of x can be a proper prefix of the description of x' (written $\gamma_X(x) \prec \gamma_X(x')$) if the symbols are *not* confusable ($x \not\doteqdot x'$) – that is, if knowing Y eliminates any ambiguity between the descriptions of x and x'. The design algorithm in [1] allows $\gamma_X(x) = \gamma_X(x')$ when $x \not\doteqdot x'$ but never allows $\gamma_X(x) \prec \gamma_X(x')$, giving a code consistent with the *unrestricted inputs* codes of [24]. While [24] and [23] don't treat code design, both address the two types of prefix violations. The discussions in [22] and [24] associate with each p.m.f. $p(x,y)$ on $\mathcal{X} \times \mathcal{Y}$ a graph $G = (\mathcal{X}, E_{\mathcal{X}})$, where there is an edge between $x, x' \in \mathcal{X}$ if and only if $x \neq x'$ and $x \doteqdot x'$. Alon and Orlitsky state that code γ_X is valid if and only if for every edge $\{x, x'\} \in E_{\mathcal{X}}$, $\{\gamma_X(x), \gamma_X(x')\}$ satisfies the prefix condition; thus a valid

code from [24] is a lossless instantaneous SISC. Since the set of uniquely decodable codes on colorings of G is a subset of the set of valid codes, they bound the expected rate of a side information code as a function of the "chromatic entropy" of G. While this approach yields elegant rate bounds, the difference between the resulting expected rate and the optimal performance can be arbitrarily large [24]. Building on the results of [24], the recent work of [33] characterizes the asymptotic rate of an SISC as the complementary graph entropy of graph G.

3.2.2 Groups, Partitions, and Matched Codes

While graphs give a strong intuition into the performance bounds of [24], I find them difficult to work with for *optimal* code design; I therefore turn instead to tree structures in the discussion that follows. I use trees to illustrate the prefix relationships between codewords: $\gamma_X(x) \prec \gamma_X(x')$ if and only if x is an ancestor of x' in the corresponding tree, and $\gamma_X(x) = \gamma_X(x')$ if and only if x and x' occupy the same node of the corresponding tree. The resulting trees are similar to Huffman code trees in that all symbols descending from a common parent have descriptions that share a common prefix; they differ from Huffman trees in that they need not be binary, symbols can reside at internal nodes as well as leaves, and multiple symbols can occupy the same node. I call each possible sub-tree a "group"; the number of levels in a group equals the number of levels in the corresponding tree. Precise definitions follow; these definitions rule out any construction that cannot yield a lossless instantaneous SISC.

The collection $\mathcal{G} = (x_1, \ldots, x_m)$ is a legitimate *1-level group* for $p(x, y)$ if for any distinct

$x_i, x_j \in \mathcal{G}$, $x_i \not\equiv x_j$.[2] The *tree representation* $\mathcal{T}(\mathcal{G})$ for 1-level group $\mathcal{G}$ is a single node representing all members of $\mathcal{G}$. For the p.m.f. in Table 6.1(a) in the Appendix, (a_0), (a_4, a_7), and (a_0, a_4, a_7) are all examples of legitimate 1-level groups.

A *2-level group* for $p(x, y)$, denoted by $\mathcal{G} = (\mathcal{R} : \mathcal{C}(\mathcal{R}))$, comprises a root $\mathcal{R}$ and its children $\mathcal{C}(\mathcal{R})$; $\mathcal{R}$ is a 1-level group, $\mathcal{C}(\mathcal{R})$ is a set of 1-level groups, and $\mathcal{G}' \not\equiv \mathcal{R}$ for all $\mathcal{G}' \in \mathcal{C}(\mathcal{R})$, where for any groups $\mathcal{G}_1$ and $\mathcal{G}_2$, $\mathcal{G}_1 \not\equiv \mathcal{G}_2$ if and only if $x_1 \not\equiv x_2$ for all $x_1 \in \mathcal{G}_1$ and $x_2 \in \mathcal{G}_2$. Members of all $\mathcal{G}' \in \mathcal{C}(\mathcal{R})$ are called members of $\mathcal{C}(\mathcal{R})$, and members of $\mathcal{R}$ and $\mathcal{C}(\mathcal{R})$ are called members of $\mathcal{G}$. In the tree representation $\mathcal{T}(\mathcal{G})$ for $\mathcal{G}$, $\mathcal{T}(\mathcal{R})$ is the root of $\mathcal{T}(\mathcal{G})$ and the parent of all subtrees $\mathcal{T}(\mathcal{G}')$ for $\mathcal{G}' \in \mathcal{C}(\mathcal{R})$. An example of a 2-level group for the p.m.f. in Table 6.1(a) is $\mathcal{G}_2 = ((a_4) : \{(a_0), (a_2, a_7), (a_6)\})$. In this case $\mathcal{R} = (a_4)$ and $\mathcal{C}(\mathcal{R}) = \{(a_0), (a_2, a_7), (a_6)\}$. The members of $\mathcal{C}(\mathcal{R})$ are $\{a_0, a_2, a_6, a_7\}$; the members of $\mathcal{G}_2$ are $\{a_0, a_2, a_4, a_6, a_7\}$. The tree representation $\mathcal{T}(\mathcal{G}_2)$ is a 2-level tree comprising a root and its three children, each of which is a single node.

For each subsequent $M > 2$, an *M-level group* for $p(x, y)$ is a pair $\mathcal{G} = (\mathcal{R} : \mathcal{C}(\mathcal{R}))$ such that $\mathcal{G}' \not\equiv \mathcal{R}$ for all $\mathcal{G}' \in \mathcal{C}(\mathcal{R})$. Here $\mathcal{R}$ is a 1-level group and $\mathcal{C}(\mathcal{R})$ is a set of groups of $M - 1$ or fewer levels, at least one of which has $(M - 1)$ levels. The members of $\mathcal{R}$ and $\mathcal{C}(\mathcal{R})$ together comprise the members of $\mathcal{G} = (\mathcal{R} : \mathcal{C}(\mathcal{R}))$. Again, $\mathcal{T}(\mathcal{R})$ is the root of $\mathcal{T}(\mathcal{G})$ and the parent of all subtrees $\mathcal{T}(\mathcal{G}')$ for $\mathcal{G}' \in \mathcal{C}(\mathcal{R})$. For any $M > 1$, an M-level group is also called a *multi-level group*. An example of a 3-level group for the p.m.f. in Table 6.1(a) is $\mathcal{G}_3 = ((a_7) : \{(a_0), (a_1), ((a_2) : \{(a_4), (a_5)\})\})$. In $\mathcal{T}(\mathcal{G}_3)$, the root $\mathcal{T}(a_7)$ of the three-level group has three children: the first two children are nodes $\mathcal{T}(a_0)$ and $\mathcal{T}(a_1)$; the third child

[2] All symbols given the same color in a coloring in [24] are a one-level group. Thus any independent set in the graph G can be a one-level group.

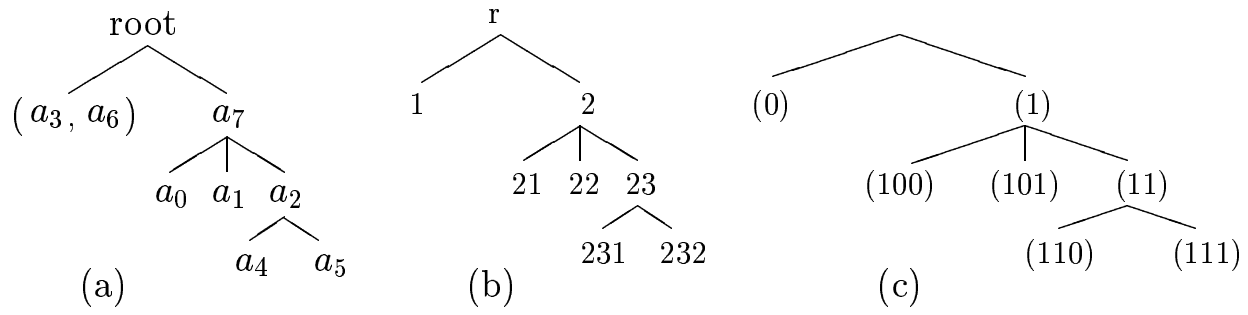


Figure 3.2: (a) Partition tree $\mathcal{T}(\mathcal{P}(\mathcal{X}))$; (b) labels for $\mathcal{T}(\mathcal{P}(\mathcal{X}))$; (c) matched code for $\mathcal{P}(\mathcal{X})$.

is a 2-level tree with root node $\mathcal{T}(a_2)$ and children $\mathcal{T}(a_4)$ and $\mathcal{T}(a_5)$.

A *partition* $\mathcal{P}(\mathcal{X})$ on $\mathcal{X}$ for p.m.f. $p(x, y)$ is a complete and non-overlapping set of groups. That is, $\mathcal{P}(\mathcal{X}) = \{\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_m\}$ satisfies $\bigcup_{i=1}^{m} \mathcal{G}_i = \mathcal{X}$ and $\mathcal{G}_j \bigcap \mathcal{G}_k = \phi$ for any $j \neq k$, where each $\mathcal{G}_i \in \mathcal{P}(\mathcal{X})$ is a group for $p(x, y)$, and $\mathcal{G}_j \cup \mathcal{G}_k$ and $\mathcal{G}_j \cap \mathcal{G}_k$ refer to the union and intersection respectively of the members of $\mathcal{G}_j$ and $\mathcal{G}_k$. The tree representation of a partition is called a *partition tree*. The partition tree $\mathcal{T}(\mathcal{P}(\mathcal{X}))$ for partition $\mathcal{P}(\mathcal{X}) = \{\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_m\}$ has an empty root $\mathbf{r}$ with m children, $\mathcal{T}(\mathcal{G}_1), \ldots, \mathcal{T}(\mathcal{G}_m)$. A partition tree is not necessarily a regular k-ary tree since the number of descendants varies from one node to the next. Figure 3.2(a) gives a partition tree for partition $\mathcal{P}(\mathcal{X}) = \{(a_3, a_6), \mathcal{G}_3\}$.

For any 1-level group $\mathcal{G}$ at depth d in $\mathcal{T}(\mathcal{P}(\mathcal{X}))$, let $\mathbf{n}$ describe the d-step path from root $\mathbf{r}$ to node $\mathcal{T}(\mathcal{G})$ in $\mathcal{T}(\mathcal{P}(\mathcal{X}))$. We often refer to $\mathcal{G}$ by describing this path. Thus $\mathcal{T}(\mathbf{n}) = \mathcal{T}(\mathcal{G})$. For notational simplicity, we sometimes substitute $\mathbf{n}$ for $\mathcal{T}(\mathbf{n})$ when it is clear from the context that we are talking about the node rather than the 1-level group at that node (e.g., $\mathbf{n} \in \mathcal{T}(\mathcal{P}(\mathcal{X}))$ rather than $\mathcal{T}(\mathbf{n}) \in \mathcal{T}(\mathcal{P}(\mathcal{X}))$). To make the path descriptions unique, I fix an order on the descendants of each node and number them from left to right. Thus $\mathbf{n}$'s children are labeled as $\mathbf{n}1, \mathbf{n}2, \ldots, \mathbf{n}K(\mathbf{n})$, where $\mathbf{n}k$ is a vector created by concatenating k

to $\mathbf{n}$ and $K(\mathbf{n})$ is the number of children descending from $\mathbf{n}$. The labeled partition tree for Figure 3.2(a) appears in Figure 3.2(b).

The *node probability* $q(\mathbf{n})$ of 1-level group $\mathbf{n}$ is the sum of the probabilities of that group's members. The *subtree probability* $Q(\mathbf{n})$ of 1-level group $\mathbf{n}$ is the sum of probabilities of $\mathbf{n}$'s members and descendants in $\mathcal{T}(\mathcal{P}(\mathcal{X}))$. In Figure 3.2(b), $q(23) = p_X(a_2)$ and $Q(23) = p_X(a_2) + p_X(a_4) + p_X(a_5)$.

A *matched code* γ_X for partition $\mathcal{P}(\mathcal{X})$ is any binary code[3] such that for any node $\mathbf{n} \in \mathcal{T}(\mathcal{P}(\mathcal{X}))$ and symbols $x_1, x_2 \in \mathbf{n}$ and $x_3 \in \mathbf{n}k$, $k \in \{1, \ldots, K(\mathbf{n})\}$: (1) $\gamma_X(x_1) = \gamma_X(x_2)$; (2) $\gamma_X(x_1) \prec \gamma_X(x_3)$; (3) $\{\gamma_X(\mathbf{n}k) : k \in \{1, \ldots, K(\mathbf{n})\}\}$ is prefix-free. (I use $\gamma_X(\mathbf{n})$ interchangeably with $\gamma_X(x)$ for any $x \in \mathbf{n}$.) If symbol $x \in \mathcal{X}$ belongs to 1-level group $\mathcal{G}$, then $\gamma_X(x)$ describes the path in $\mathcal{T}(\mathcal{P}(\mathcal{X}))$ from $\mathbf{r}$ to $\mathcal{T}(\mathcal{G})$; the path description is a concatenated list of step descriptions, where the step from $\mathbf{n}$ to $\mathbf{n}k$, $k \in \{1, \ldots, K(\mathbf{n})\}$, is described using a prefix-code on $\{1, \ldots, K(\mathbf{n})\}$.

An example of a matched code for the partition of Figure 3.2(a) appears in Figure 3.2(c). Figure 3.3 shows that code's encoder and decoder.

In the decoder definition '$-$' marks an event that can never occur, '$\downarrow$' marks a case where the decoder knows it has not reached the end of $\gamma_X(x)$, and '$\uparrow$' marks a case where the decoder would have decoded based on fewer symbol than are given. For example, since only $\gamma_X(a_3)$ and $\gamma_X(a_6)$ begin with $0\ldots$ and $p(a_3, a_0) = p(a_6, a_0) = 0$, the decoder never receives $\gamma_X(x) = 0\cdots$ when $Y = a_0$; if the decoder receives binary string $11\ldots$ when $Y = a_1$, then the decoder has not reached the end of $\gamma_X(x_1)$ since $p(a_7, a_1) = p(a_2, a_1) = 0$

[3] I here focus on codes with binary channel alphabet $\{0, 1\}$. The extension to codes with other finite channel alphabets is straightforward.

$$\gamma_X(x) = \begin{cases} 0 & \text{if } x \in \{a_3, a_6\} \\ 1 & \text{if } x = a_7 \\ 100 & \text{if } x = a_0 \\ 101 & \text{if } x = a_1 \\ 11 & \text{if } x = a_2 \\ 110 & \text{if } x = a_4 \\ 111 & \text{if } x = a_5 \end{cases}$$

	$\gamma^{-1}((\gamma_X(x_1)\ldots), y)$							
$\gamma_X(x_1)\ldots \backslash y$	a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7
$0\ldots$	$-$	a_3	a_6	a_3	$-$	a_6	a_6	a_6
$1\ldots$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	a_7	a_7
$10\ldots$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\uparrow$	$\uparrow$
$100\ldots$	a_0	$-$	a_0	a_0	$-$	$-$	$\uparrow$	$\uparrow$
$101\ldots$	$-$	a_1	$-$	$-$	a_1	a_1	$\uparrow$	$\uparrow$
$11\ldots$	a_2	$\downarrow$	a_2	$\downarrow$	$\downarrow$	$\downarrow$	$\uparrow$	$\uparrow$
$110\ldots$	$\uparrow$	a_4	$\uparrow$	$-$	a_4	$-$	$\uparrow$	$\uparrow$
$111\ldots$	$\uparrow$	$-$	$\uparrow$	a_5	a_5	a_5	$\uparrow$	$\uparrow$

Figure 3.3: The encoder and decoder for the example in Figure 3.2. For the decoder, $-$ marks an event that can never occur, $\downarrow$ marks a case where the decoder knows it has not reached the end of $\gamma_X(x)$, and $\uparrow$ marks a case where the decoder would have decoded based on fewer symbol than are given.

and every other $\gamma_X(x_1)$ beginning $11\ldots$ has length greater than 2; if the decoder receives binary string $111\ldots$ when $Y = a_0$, then it decodes to a_2 after only 2 bits. The code is instantaneous since the decoder can always decode after no more than $|\gamma_X(x)|$ bits. The code is lossless since $\gamma^{-1}(\gamma_X(x), y) = x$ whenever $p(x, y) > 0$ and probability 0 events cannot occur. The code achieves expected rate $E_X|\gamma_X(X)| = 2.12 < H(X) = 2.91$ by violating Kraft's inequality. (Kraft's inequality does not apply to SISCs [23].) This rate may be further reduced if we choose the partition and its matched code more carefully. For example, setting $\gamma_X(a_0) = \gamma_X(a_1) = 10$ in this code gives a lossless instantaneous code with lower expected rate.

3.2.3 All Lossless Codes are Matched Codes

In the above framework, a partition specifies the prefix and equivalence relationships in the binary descriptions of $x \in \mathcal{X}$; a matched code is any code with those properties. Theorem 2 establishes the equivalence of matched codes and lossless instantaneous SISCs.

Theorem 2 *Code γ_X is a lossless, instantaneous SISC for $p(x, y)$ if and only if γ_X is a matched code for some partition $\mathcal{P}(\mathcal{X})$ for $p(x, y)$.*

Proof: Forward: By the definitions of partitions and matched codes, only symbols that are not confusable can be assigned codewords that violate the prefix condition. Thus any matched code is a lossless instantaneous code by Lemma 1.

Converse: Given γ_X, we construct a partition $\mathcal{P}(\mathcal{X})$ for $p(x, y)$ such that γ_X is a matched code for $\mathcal{P}(\mathcal{X})$. We begin by building a binary tree $\mathcal{T}_2$ with symbol x at the node reached by following path $\gamma_X(x)$ downward from the tree root. We build partition tree $\mathcal{T}$ from binary

tree $\mathcal{T}_2$ by visiting the nodes of $\mathcal{T}_2$ one by one and modifying them as follows. If the current node is the root of the tree and that node is occupied by some $x \in \mathcal{X}$, we add a new node $\mathbf{r}$ as the parent of the current node; if the current node is not the root of the tree and that node is empty, we remove the current node from the tree, attaching the node's children (if any) directly to the node's parent; otherwise, we make no change. Tree $\mathcal{T}$ is a partition tree for some partition $\mathcal{P}(\mathcal{X})$ for $p(x, y)$ since: (1) γ_X lossless implies each non-empty node $\mathbf{n} \in \mathcal{T}$ is a legitimate 1-level group; (2) γ_X instantaneous implies each subtree of $\mathcal{T}$ is a legitimate multi-level group; and (3) the root of $\mathcal{T}$ is an empty node with one or more multi-level groups descending from it. Here (1) and (2) follow from Lemma 1 while (3) is by construction given (1) and (2). Code γ_X is a matched code for $\mathcal{P}(\mathcal{X})$ since: for any $\mathbf{n} \in \mathcal{T}(\mathcal{P}(\mathcal{X}))$, $x_1, x_2 \in \mathbf{n}$, and $x_3 \in \mathbf{n}k$, $\gamma_X(x_1) = \gamma_X(x_2)$, $\gamma_X(x_1) \prec \gamma_X(x_3)$; and $\{\gamma_X(\mathbf{n}k) : k \in \{1, \ldots, K(\mathbf{n})\}\}$ is prefix-free by construction. □

Using Theorem 2, I break the problem of lossless SISC design into two parts: partition design and matched code design. While the choice of 1-level groups in a partition design is equivalent to choosing a coloring of the nodes of graph G, the coloring corresponding to the optimal partition need not be the entropy-minimizing coloring [24]. I conjecture that both finding the optimal coloring and finding the optimal prefix relationships for that coloring are NP-hard problems, [4] and I combine these "hard" parts into the single step of partition

[4] The intuition behind the first conjecture results from the difficulty of coloring problems for a wide variety of applications. The intuition for the second conjecture comes from the observation that finding the optimal prefix relationship is an optimal partition design problem for the set of distributions whose optimal partitions use proper prefix relationships but do not allow $\gamma_X(x) = \gamma_X(x')$ for any $x \neq x'$. Since this restriction does not suggest any obvious constraints on $p(x, y)$, I conjecture that finding optimal prefix relationships, like optimal partition design, is NP-hard.

design. I treat both optimal partition design and fast approximation algorithms in later sections. Given a partition, optimal matched code design requires only polynomial time. I treat optimal matched code design next. There is no analogue to matched code design in [24].

3.2.4 Matched Code Design: Optimal Huffman and Arithmetic Codes

I wish to design the optimal matched code for an arbitrary fixed partition $\mathcal{P}(\mathcal{X})$ for $p(x, y)$. In traditional lossless coding, the optimal description lengths are $l^\star(x) = -\log p(x)$ for all $x \in \mathcal{X}$ if those lengths are all integers. Theorem 3 gives the corresponding result for lossless SISCs on a fixed partition $\mathcal{P}(\mathcal{X})$.

Theorem 3 *Given partition $\mathcal{P}(\mathcal{X})$ for $p(x, y)$, the optimal matched code for $\mathcal{P}(\mathcal{X})$ has description lengths $l^\star(\mathbf{r}) = 0$ and*

$$l^\star(\mathbf{n}k) = l^\star(\mathbf{n}) - \log_2 \left(\frac{Q(\mathbf{n}k)}{\sum_{j=1}^{K(\mathbf{n})} Q(\mathbf{n}j)} \right)$$

for all $\mathbf{n} \in \mathcal{T}(\mathcal{P}(\mathcal{X}))$ and $k \in \{1, \ldots, K(\mathbf{n})\}$ if those lengths are all integers. Here $l^\star(\mathbf{n}) = l$ implies $l^\star(x) = l$ for each symbol x in 1-level group $\mathbf{n}$.

Proof: Given $x \in \mathbf{n} \in \mathcal{T}(\mathcal{P}(\mathcal{X}))$, let $l(\mathbf{n}) = |\gamma_X(x)|$. Then for any matched code γ_X for $\mathcal{P}(\mathcal{X})$,

$$E|\gamma_X(X)| = \sum_{\mathbf{n} \in \mathcal{T}(\mathcal{P}(\mathcal{X}))} q(\mathbf{n}) l(\mathbf{n}) = \sum_{\mathbf{n} \in \mathcal{T}(\mathcal{P}(\mathcal{X}))} \sum_{k=1}^{K(\mathbf{n})} Q(\mathbf{n}k)(l(\mathbf{n}k) - l(\mathbf{n})).$$

Thus the minimal expected rate is achieved by minimizing each sum $\sum_{k=1}^{K(\mathbf{n})} Q(\mathbf{n}k)(l(\mathbf{n}k) - l(\mathbf{n}))$ independently. If we write $\gamma_X(\mathbf{n}k) = g g_k$ for each $k \in \{1, \ldots K(\mathbf{n})\}$, where $g = \gamma_X(\mathbf{n})$,

then g_k is the suffix associated with the kth descendant of $\mathbf{n}$ and $|g_k| = l(\mathbf{n}k) - l(\mathbf{n})$. For $\{g_1, \ldots, g_{K(\mathbf{n})}\}$ to satisfy the prefix condition, $\{|g_1|, \ldots, |g_{K(\mathbf{n})}|\}$ must satisfy Kraft's inequality. The minimization of $\sum_{k=1}^{K(\mathbf{n})} Q(\mathbf{n}k)|g_k|$ subject to $\sum_{k=1}^{K(\mathbf{n})} 2^{-|g_k|} \leq 1$ is achieved by setting

$$|g_k| = l(\mathbf{n}k) - l(\mathbf{n}) = -\log_2 \left(\frac{Q(\mathbf{n}k)}{\sum_{j=1}^{K(\mathbf{n})} Q(\mathbf{n}j)} \right),$$

which completes the proof. □

The proof of Theorem 3 demonstrates that we can design matched codes by designing entropy codes on the children of each internal node of a partition tree. All entropy coding algorithms are candidates for matched code design. I focus on matched Huffman and arithmetic coding. For any node $\mathbf{n}$ with $K(\mathbf{n}) > 0$, the Huffman code $\gamma^{(H)}_{X,\mathcal{P}(\mathcal{X})}$ describes the step from $\mathbf{n}$ to $\mathbf{n}k$ using a Huffman code designed for p.m.f. $\{Q(\mathbf{n}k)/\sum_{j=1}^{K(\mathbf{n})} Q(\mathbf{n}j)\}_{k=1}^{K(\mathbf{n})}$ on alphabet $\{1, \ldots, K(\mathbf{n})\}$. The arithmetic code $\gamma^{(A)}_{X,\mathcal{P}(\mathcal{X})}$ uses arithmetic codes matched to the same p.m.f.s. Both of these strategies give polynomial time algorithms. Theorem 4 proves the optimality of matched Huffman codes. Before giving that result, I give an example.

Example: In building a matched Huffman code for the partition in Figure 3.2(a), we work from the top of partition tree $\mathcal{T}$. We begin by designing a Huffman code for p.m.f. $\{Q(k)/\sum_{j=1}^{K(\mathbf{r})} Q(j)\}_{k=1}^{K(\mathbf{r})}$ on the $K(\mathbf{r})$ descendants of the (empty) root of $\mathcal{T}$. In this case, $K(\mathbf{r}) = 2$ ($'1' = (a_3, a_6)$ and $'2' = (a_7)$), the p.m.f. is $\{p_X(a_3) + p_X(a_6), p_X(a_7) + p_X(a_0) + p_X(a_1) + p_X(a_2) + p_X(a_4) + p_X(a_5)\} = \{.21, .79\}$, and the Huffman code is $\{0, 1\}$. We repeat this process for each subsequent tree node $\mathbf{n}$ with $K(\mathbf{n}) > 0$. Node 2 gives $K(2) = 3$, p.m.f. $\{p_X(a_0)/Q_1, p_X(a_1)/Q_1, (p_X(a_2) + p_X(a_4) + p_X(a_5))/Q_1\} = \{.1/Q_1, .19/Q_1, .37/Q_1\}$ ($Q_1 = .66$), and Huffman code $\{00, 01, 1\}$. Node 23 gives $K(23) = 2$, p.m.f. $\{p_X(a_4)/Q_2, p_X(a_5)/Q_2\}$

$= \{.11/Q_2, .06/Q_2\}$ ($Q_2 = .17$), and Huffman code $\{0, 1\}$. Finally, $\gamma_X(\mathbf{n})$ concatenates the Huffman codewords for all branches traversed in moving from $\mathbf{r}$ to $\mathbf{n}$ in $\mathcal{T}$. The codewords for this example appear in Figure 3.2(c).

Theorem 4 *Given a partition $\mathcal{P}(\mathcal{X})$, matched Huffman codes for $\mathcal{P}(\mathcal{X})$ achieve the optimal expected rate over all matched codes for $\mathcal{P}(\mathcal{X})$.*

Proof: Let $\mathcal{T}$ be the partition tree of $\mathcal{P}(\mathcal{X})$. The codelength of a node $\mathbf{n} \in \mathcal{T}$ is denoted by $l(\mathbf{n})$. The average length $\bar{l}$ for $\mathcal{P}(\mathcal{X})$ is

$$\bar{l} = \sum_{\mathbf{n}\in\mathcal{T}} q(\mathbf{n})l(\mathbf{n}) = \sum_{k=1}^{K(\mathbf{r})} \left(Q(k)l(k) + \Delta\bar{l}(k)\right),$$

where for each $k \in \{1, \ldots, K(\mathbf{r})\}$, $\Delta\bar{l}(k) = \sum_{k\mathbf{n}\in\mathcal{T}} q(k\mathbf{n})(l(k\mathbf{n}) - l(k))$.

Note that $\sum_{k=1}^{K(\mathbf{r})} Q(k)l(k)$ and $\{\Delta\bar{l}(k)\}$ can be minimized independently. Thus

$$\min \bar{l} = \min \sum_{k=1}^{K(\mathbf{r})} Q(k)l(k) + \sum_{k=1}^{K(\mathbf{r})} \min \Delta\bar{l}(k).$$

In matched Huffman coding, working from the top to the bottom of the partition tree, we first minimize $\sum_{k=1}^{K(\mathbf{r})} Q(k)l(k)$ over all integer lengths $l(k)$ by employing Huffman codes on $Q(k)$. We then minimize each $\Delta\bar{l}(k)$ over all integer length codes by similarly breaking each down layer by layer and minimizing the expected length at each layer. □

In traditional arithmetic coding (with no side information), the description length of data sequence x^n is $l(x^n) = \lceil -\log p_X(x^n) \rceil + 1$, where $p_X(x^n)$ is the probability of x^n. In designing the matched arithmetic code for x^n for a given partition $\mathcal{P}(\mathcal{X})$, we use the decoder's knowledge of y^n to decrease the description length of x^n. The following example, illustrated in Figure 3.4, demonstrates the techniques of matched arithmetic coding for the partition given in Figure 3.2(a).

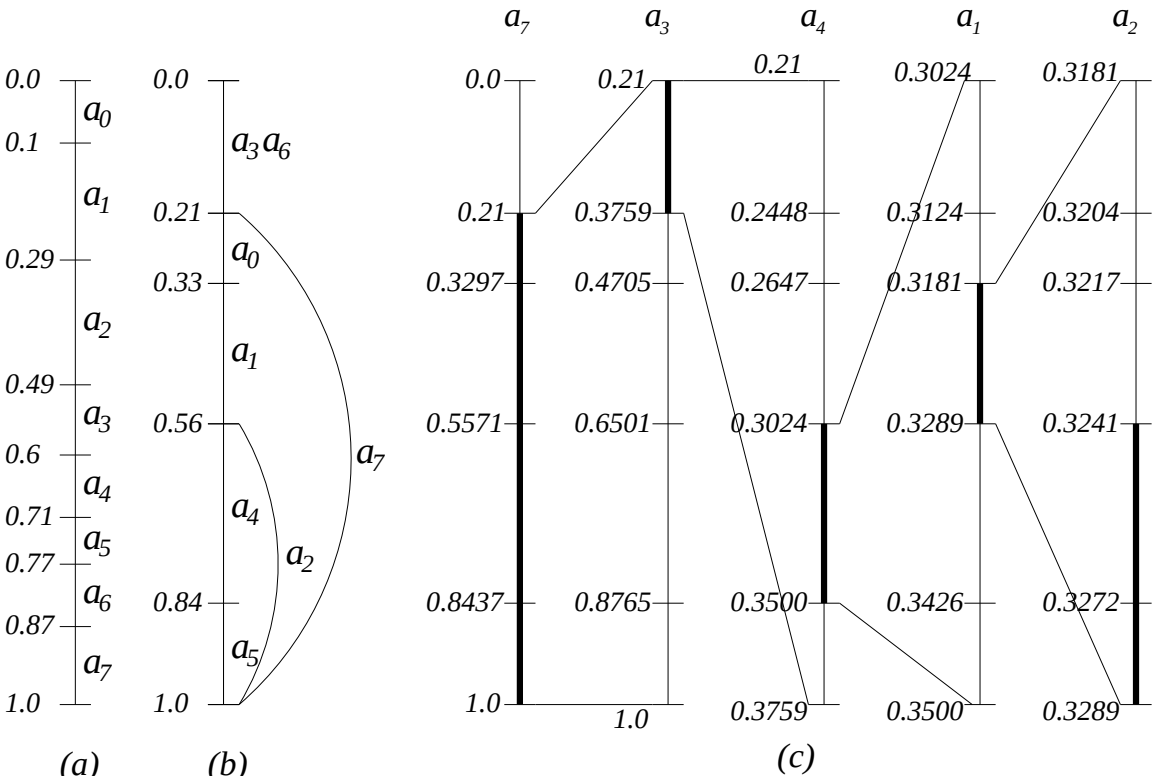


Figure 3.4: Dividing the unit interval in (a) traditional arithmetic coding and (b) matched arithmetic coding for partition $\mathcal{P}(\mathcal{X})$ of Figure 3.2(a). (c) Matched arithmetic coding for sequence $a_7a_3a_4a_1a_2$.

In traditional arithmetic coding, data sequence X^n is represented by an interval of the $[0,1)$ line. We describe X^n by describing the mid-point of the corresponding interval to sufficient accuracy to avoid confusion with neighboring intervals. We find the interval for x^n recursively. We first break $[0,1)$ into intervals corresponding to all possible values of x_1 (see Figure 3.4(a)). Then we break the interval for the observed X_1 into subintervals corresponding to all possible values of X_1x_2, and so on. Given the interval $A \subseteq [0,1)$ for X^k for some $0 \leq k < n$ (the interval for X^0 is $[0,1)$), the subintervals for $\{X^k x_{k+1}\}$ are ordered subintervals of A with lengths proportional to $p(x_{k+1})$.

In matched arithmetic coding for partition $\mathcal{P}(\mathcal{X})$, we again describe X^n by describing the mid-point of a recursively constructed subinterval of $[0,1)$. The intervals here correspond to nodes, and we describe symbol $x \in \mathcal{X}$ by describing the mid-point of the interval corresponding to the node $\mathbf{n}$ for which $x \in \mathbf{n}$. In describing x_1, the interval for root $\mathbf{r}$ is $[0,1)$ with length $p^{(A)}(\mathbf{r}) = 1$. We define the remainder of the intervals recursively. The interval for any $\mathbf{n} \in \mathcal{T}(\mathcal{P}(\mathcal{X}))$ comprises $K(\mathbf{n})$ ordered subintervals of lengths

$$p^{(A)}(\mathbf{n}k) = \left(\frac{Q(\mathbf{n}k)}{\sum_{k=1}^{K(\mathbf{n})} Q(\mathbf{n}k)}\right) p^{(A)}(\mathbf{n}) = \left(\frac{Q(\mathbf{n}k)}{Q(\mathbf{n}) - q(\mathbf{n})}\right) p^{(A)}(\mathbf{n}) \qquad k \in \{1, \ldots, K(\mathbf{n})\},$$

corresponding to the $K(\mathbf{n})$ children of $\mathbf{n}$ in the partition tree. The nested nature of the intervals parallels the situation in matched Huffman coding where one symbol's description is the prefix of another symbol's description. Again, for any legitimate partition $\mathcal{P}(\mathcal{X})$, the decoder can uniquely distinguish between symbols with nested intervals using its knowledge of the side information.

Refining the interval for sequence X^{i-1} to find the subinterval for X^i involves carving the current interval into subintervals of sizes proportional to those found above. We finally

describe X^n by describing the center of its corresponding subinterval to an accuracy sufficient to distinguish it from its neighboring subintervals. To ensure unique decodability,

$$l^{(A)}(x^n) = \lceil -\log p^{(A)}(x^n) \rceil + 1,$$

where $p^{(A)}(x^n)$ is the length of the subinterval corresponding to string x^n. Given a fixed partition $\mathcal{P}(\mathcal{X})$, suppose $x \in \mathbf{n}(x) \in \mathcal{T}(\mathcal{P}(\mathcal{X}))$ and $\mathbf{n}_0(x)$ is the parent of $\mathbf{n}(x)$. Then

$$\begin{aligned} l^{(A)}(x^n) &= \lceil -\log p^{(A)}(x^n) \rceil + 1 \\ &= \left\lceil \sum_{i=1}^{n} -\log p^{(A)}(\mathbf{n}(x_i)) \right\rceil + 1 \\ &= \left\lceil \sum_{i=1}^{n} \left(-\log p^{(A)}(\mathbf{n}_0(x_i)) - \log \frac{Q(\mathbf{n}(x_i))}{\sum_{k=1}^{K(\mathbf{n}_0(x_i))} Q(\mathbf{n}_0(x_i)k)} \right) \right\rceil + 1 \\ &< \sum_{i=1}^{n} l^{\star}(x_i) + 2 \end{aligned}$$

where $l^{\star}()$ is the optimal length function specified in Theorem 3. Thus the description length $l^{(A)}(x^n)$ in coding data sequence x^n using a 1-dimensional "matched arithmetic code" $\gamma^{(A)}_{X,\mathcal{P}(\mathcal{X})}$ satisfies $(1/n)l^{(A)}(x^n) < (1/n)\sum_{i=1}^{n} l^{\star}(x_i) + 2/n$, giving a normalized description length arbitrarily close to the one-dimensional optimum for n sufficiently large. (In practice, assuming that a large number of symbols are coded, we can always use $El^{\star}_{\mathcal{P}(\mathcal{X})}(X)$ as the rate for arithmetic coding, neglecting as trivial the $2/n$ term in the above bound.) We deal with floating point precision issues using the same techniques applied to traditional arithmetic codes.

Example: Again consider the p.m.f. of Table 6.1(a) and the partition of Figure 3.2(a). The interval for the root is $[0, 1)$ with subintervals $[0, .21)$ and $[.21, 1)$ for children $'1' = (a_3, a_6)$ and $'2' = (a_7)$, respectively. Interval $[.21, 1)$ comprises subintervals $[.21, .3297)$, $[.3297, .5571)$,

and $[.5571, 1)$ for $'21' = (a_0)$, $'22' = (a_1)$, and $'23' = (a_2)$ since

$$\begin{aligned} p^{(A)}((a_0)) &= p^{(A)}((a_7))\frac{Q((a_0))}{Q((a_7)) - q((a_7))} = .79\frac{.1}{.79 - .13} = .1197 \\ p^{(A)}((a_1)) &= p^{(A)}((a_7))\frac{Q((a_1))}{Q((a_7)) - q((a_7))} = .79\frac{.19}{.79 - .13} = .2274 \\ p^{(A)}((a_2)) &= p^{(A)}((a_7))\frac{Q((a_2))}{Q((a_7)) - q((a_7))} = .79\frac{.37}{.79 - .13} = .4428. \end{aligned}$$

Interval $[.5571, 1)$ comprises subintervals $[.5571, .8437)$ and $[.8437, 1)$ for $'231' = (a_4)$ and $'232' = (a_5)$ since

$$\begin{aligned} p^{(A)}((a_4)) &= p^{(A)}((a_2))\frac{Q((a_4))}{Q((a_2)) - q((a_2))} = .4428(.11/(.37 - .2)) = .2866 \\ p^{(A)}((a_5)) &= p^{(A)}((a_2))\frac{Q((a_5))}{Q((a_2)) - q((a_2))} = .4428(.06/(.37 - .2)) = .1563. \end{aligned}$$

Figure 3.4(b) shows these intervals.

Figure 3.4(c) shows the recursive interval refinement procedure for $X^5 = (a_7 a_3 a_4 a_1 a_2)$. Symbol $X_1 = a_7$ gives interval $[0.21, 1)$ of length .79 (indicated by the bold line). Symbol $X_2 = a_3$ refines the above interval to the interval $[.21, .3759)$ of length $.21 \cdot .79 = .1659$. Symbol $X_3 = a_4$ refines that interval to the interval $[.3024, .3500)$ of length $.2866 \cdot .1659 = .0475$. This procedure continues, giving final interval $[0.3241, 0.3289)$.

3.2.5 Optimal Partitions: Definitions and Properties

The preceding discussion treats matched code design for a given partition $\mathcal{P}(\mathcal{X})$. The partition yielding the best performance remains to be found.

Given a partition $\mathcal{P}(\mathcal{X})$, let $l^{(H)}_{\mathcal{P}(\mathcal{X})}$ and $l^{\star}_{\mathcal{P}(\mathcal{X})}$ be the Huffman and optimal description lengths respectively for $\mathcal{P}(\mathcal{X})$. We say that $\mathcal{P}(\mathcal{X})$ is *optimal for a matched Huffman SISC* on $p(x, y)$ if $El^{(H)}_{\mathcal{P}(\mathcal{X})}(X) \leq El^{(H)}_{\mathcal{P}'(\mathcal{X})}(X)$ for any partition $\mathcal{P}'(\mathcal{X})$ for $p(x, y)$ (and therefore, by

Theorems 2 and 4, $El^{(H)}_{\mathcal{P}(\mathcal{X})}(X) \leq El(X)$ where l is the description length for any instantaneous lossless SISC on $p(x,y)$). We say that $\mathcal{P}(\mathcal{X})$ is *optimal for a matched arithmetic SISC* on $p(x,y)$ if $El^{\star}_{\mathcal{P}(\mathcal{X})}(X) \leq El^{\star}_{\mathcal{P}'(\mathcal{X})}(X)$ for any partition $\mathcal{P}'(\mathcal{X})$ for $p(x,y)$ since the arithmetic code approaches the optimal one-dimensional expected rate of Theorem 3 as n (the number of symbols coded) grows.

Some properties of optimal partitions follow.

Lemma 2 *There exists an optimal partition $\mathcal{P}^{\star}(\mathcal{X})$ for $p(x,y)$ for which every node except for the root of $\mathcal{P}^{\star}(\mathcal{X})$ is non-empty and no node except for the root can have exactly one child.*

Proof: If any non-root node $\mathbf{n}$ of partition $\mathcal{P}(\mathcal{X})$ is empty, then removing $\mathbf{n}$, so $\{\mathbf{n}k\}_{k=1}^{K(\mathbf{n})}$ descend directly from $\mathbf{n}$'s parent, gives new partition $\mathcal{P}'(\mathcal{X})$. Any matched code on $\mathcal{P}(\mathcal{X})$, including the optimal matched code on $\mathcal{P}(\mathcal{X})$, is a matched code on $\mathcal{P}'(\mathcal{X})$. If $\mathbf{n}$ has exactly one child, then combining $\mathbf{n}$ and its child yields a legitimate partition $\mathcal{P}'(\mathcal{X})$; the optimal matched code for $\mathcal{P}'(\mathcal{X})$ yields expected rate no worse than that of the optimal matched code for $\mathcal{P}(\mathcal{X})$. □

Lemma 3 *Let $\mathcal{T}(\mathbf{n})$ be an arbitrary node in optimal partition $\mathcal{P}^{\star}(\mathcal{X})$ for $p(x,y)$, and let $\mathcal{G} = ((\mathbf{n}) : \mathcal{C}(\mathbf{n}))$ be the group with root $\mathbf{n}$ and descendants identical to the descendants of $\mathbf{n}$ in $\mathcal{P}^{\star}(\mathcal{X})$. Then $\mathbf{n} = \{x \in \mathcal{G} : \{x\} \neq (\mathcal{G} \cap \{x\}^c)\}$ and $\mathcal{C}(\mathbf{n})$ is an optimal partition of $\{x \in \mathcal{G} : x \notin \mathbf{n}\}$.*

Proof: Since the matched code's description can be broken into a description of $\mathbf{n}$ followed by a matched code on $\mathcal{C}(\mathbf{n})$ and the corresponding description lengths add, the partition described by $\mathcal{T}(\mathcal{P}(\mathcal{X}))$ cannot be optimal unless the partition described by $\mathcal{C}(\mathbf{n})$ is. □

Lemma 4 *The optimal partitions for matched Huffman and arithmetic SISCs can differ.*

Proof: I give a proof by example. For the p.m.f. of Table 6.1(a), the optimal partition for a matched Huffman SISC is $\{(a_0, a_1), ((a_2, a_7) : \{(a_3), (a_5)\}), (a_4, a_6)\}$ while the optimal partition for a matched arithmetic SISC is $\{(a_3, a_6), ((a_7) : \{(a_0, a_4), ((a_2) : \{(a_1)(a_5)\})\})\}$.

□

Lemmas 2 and 3 apply to both cases. Lemma 4 results since Huffman codes use true single-symbol coding ($n = 1$) while arithmetic codes minimize the expected rate when n is large. The rates are related as

$$\lim_{n\to\infty} \frac{1}{n} El^{\star}_{\mathcal{P}^{(A)}(\mathcal{X})}(X^n) \leq El^{(H)}_{\mathcal{P}^{(H)}(\mathcal{X})}(X) < \lim_{n\to\infty} \frac{1}{n} El^{\star}_{\mathcal{P}^{(A)}(\mathcal{X})}(X^n) + 1.$$

I next show that there exist pairs of groups $(\mathcal{G}_I, \mathcal{G}_J)$ such that $\mathcal{G}_I \cap \mathcal{G}_J = \emptyset$ but $\mathcal{G}_I$ and $\mathcal{G}_J$ cannot both descend from the root of an optimal partition. This result is derived by showing conditions under which there exists a group $\mathcal{G}^{\star}$ that combines the members of $\mathcal{G}_I$ and $\mathcal{G}_J$ and for which replacing $\{\mathcal{G}_I, \mathcal{G}_J\}$ with $\{\mathcal{G}^{\star}\}$ in $\mathcal{P}(\mathcal{X})$ guarantees a performance improvement.

Suppose that $\mathcal{G}_I, \mathcal{G}_J \in \mathcal{P}(\mathcal{X})$, so that $\mathcal{G}_I$ and $\mathcal{G}_J$ extend directly from the root $\mathbf{r}$ of $\mathcal{T}(\mathcal{P}(\mathcal{X}))$ and nodes I and J are the roots of $\mathcal{T}(\mathcal{G}_I)$ and $\mathcal{T}(\mathcal{G}_J)$. Let $\mathbf{n}_\circ$ denote the 1-level group at some node in $\mathcal{T}(\mathcal{G}_J)$. We say that $\mathcal{G}_I$ *can be combined with* $\mathcal{G}_J$ *at* $\mathbf{n}_\circ$ if the combined group $\mathcal{G}^{\star}$ is a legitimate group, where $\mathcal{G}^{\star}$ is obtained by replacing $\mathbf{n}_\circ$ with 1-level group $(I, \mathbf{n}_\circ)$ and adding the descendants of I (in addition to the descendants of $\mathbf{n}_\circ$) as descendants of $(I, \mathbf{n}_\circ)$ in $\mathcal{T}(\mathcal{G}^{\star})$. Figures 3.5(a) and (b) show an example where groups $\mathcal{G}_I = ((a_2) : \{(a_4), (a_5)\})$ and $\mathcal{G}_J = ((a_7) : \{(a_1), (a_3)\})$ of partition $\mathcal{P}(\mathcal{X}) = \{(a_0), (a_6), \mathcal{G}_I, \mathcal{G}_J\}$ combine at (a_2). The modified partition is $\mathcal{P}^{\star}(\mathcal{X}) = \{(a_0), (a_6), \mathcal{G}^{\star}\}$, where $\mathcal{G}^{\star} = ((a_2, a_7) : \{(a_1), (a_3), (a_4), (a_5)\})$.

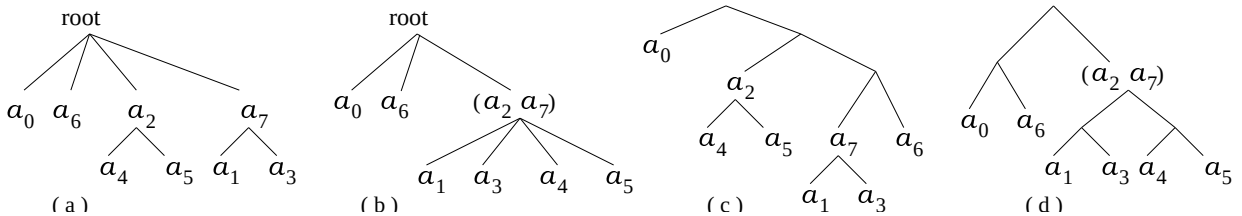


Figure 3.5: Combining the groups in partition (a) to get partition (b); (c) the matched Huffman code rate for (a) is 2.25; (d) the matched Huffman code rate for (b) is 2.3.

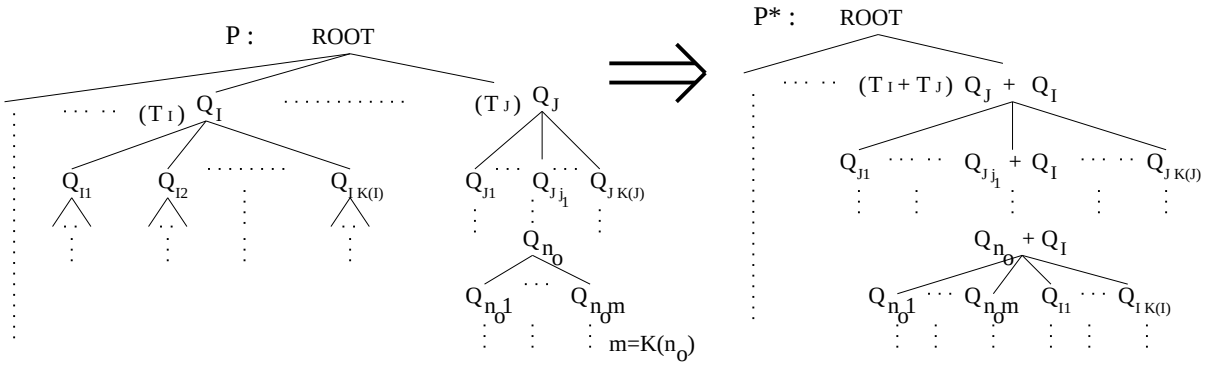


Figure 3.6: Combining two groups ($\mathcal{G}_I$ and $\mathcal{G}_J$) into one group.

Theorem 5 *Let $\mathcal{P}(\mathcal{X}) = \{\mathcal{G}_1, \ldots, \mathcal{G}_m\}$ be a partition of $\mathcal{X}$ under $p(x, y)$. Suppose that $\mathcal{G}_I \in \mathcal{P}(\mathcal{X})$ can be combined with $\mathcal{G}_J \in \mathcal{P}(\mathcal{X})$ at 1-level group $\mathbf{n}_\circ$ in $\mathcal{T}(\mathcal{G}_J)$. Let $\mathcal{P}^\star(\mathcal{X})$ be the resulting partition. Then $El^\star_{\mathcal{P}^\star(\mathcal{X})}(X) \le El^\star_{\mathcal{P}(\mathcal{X})}(X)$.*

Proof: Let $\mathbf{n}_\circ = Jj_1 \ldots j_M = \mathbf{n}_p j_M$, so that $\mathbf{n}_p$ is the parent of $\mathbf{n}_\circ$. Let $\mathcal{S}_1 = \{Jj_1 \ldots j_i : 1 \le i \le M\}$ be all nodes on the path to $\mathbf{n}_\circ$ except node J, $\mathcal{S}_2 = \{\mathbf{n} \in \mathcal{T}(\mathcal{G}_J) : \mathbf{n}$ is the sibling of node $\mathbf{s} \in \mathcal{S}_1\}$ be their siblings, and $\mathcal{S}_3 = (\mathcal{S}_1 \cup \{J\}) \cap \{\mathbf{n}_\circ\}^c$ be $\mathcal{S}_1$ modified to include J and exclude $\mathbf{n}_\circ$. Let $Q_{\mathbf{n}}$ and $q_{\mathbf{n}}$ denote the subtree and node probabilities respectively for any node $\mathbf{n} \in \mathcal{T}(\mathcal{P}(\mathcal{X}))$, and define $\Delta Q_{\mathbf{n}} = Q_{\mathbf{n}} - q_{\mathbf{n}} = \sum_{j=1}^{K(\mathbf{n})} Q_{\mathbf{n}j}$. Then Figure 3.6 shows the subtree probabilities associated with combining $\mathcal{G}_I$ with $\mathcal{G}_J$ at $\mathbf{n}_\circ$. Let the resulting new group be $\mathcal{G}^\star$.

The sum of the subtree probabilities of $\mathcal{G}_I$ and $\mathcal{G}_J$ equals the subtree probability of $\mathcal{G}^\star$, and thus the optimal average rate of the groups in $\mathcal{P}(\mathcal{X}) \cap \{\mathcal{G}_I, \mathcal{G}_J\}^c$ are not changed by the combination. Thus if $(\bar{l}_I, \bar{l}_J)$ and $(\bar{l}_I^\star, \bar{l}_J^\star)$ are the optimal average rates for $(\mathcal{G}_I, \mathcal{G}_J)$ in $\mathcal{P}(\mathcal{X})$ and $\mathcal{P}^\star(\mathcal{X})$, respectively, then $\Delta\bar{l}_I + \Delta\bar{l}_J = (\bar{l}_I - \bar{l}_I^\star) + (\bar{l}_J - \bar{l}_J^\star)$ gives the total rate cost of using partition $\mathcal{P}(\mathcal{X})$ rather than partition $\mathcal{P}^\star(\mathcal{X})$. From Theorem 3 we have

$$\begin{aligned}
-\bar{l}_I &= Q_I \log Q_I + \sum_{k=1}^{K(I)} Q_{Ik} \log \frac{Q_{Ik}}{\Delta Q_I} + \Delta l_I \\
-\bar{l}_I^\star &= Q_I \log \left((Q_I + Q_J) \prod_{\mathbf{n}k \in \mathcal{S}_1} \frac{Q_I + Q_{\mathbf{n}k}}{Q_I + \Delta Q_{\mathbf{n}}} \right) + \sum_{k=1}^{K(I)} Q_{Ik} \log \frac{Q_{Ik}}{\Delta Q_I + \Delta Q_{\mathbf{n}_\circ}} + \Delta l_I
\end{aligned}$$

which implies

$$\begin{aligned}
\Delta\bar{l}_I &= Q_I \log \left(\frac{Q_I + Q_J}{Q_I} \prod_{\mathbf{n}k \in \mathcal{S}_1} \frac{Q_I + Q_{\mathbf{n}k}}{Q_I + \Delta Q_{\mathbf{n}}} \right) + \sum_{k=1}^{K(I)} Q_{Ik} \log \frac{\Delta Q_I}{\Delta Q_I + \Delta Q_{\mathbf{n}_\circ}} \\
&= Q_I \log \left(\frac{Q_I + Q_J}{Q_I + \Delta Q_J} \frac{Q_I + Q_{Jj_1}}{Q_I + \Delta Q_{Jj_1}} \cdots \frac{Q_I + Q_{\mathbf{n}_p}}{Q_I + \Delta Q_{\mathbf{n}_p}} \frac{Q_I + Q_{\mathbf{n}_\circ}}{Q_I} \right) + \Delta Q_I \log \frac{\Delta Q_I}{\Delta Q_I + \Delta Q_{\mathbf{n}_\circ}} \\
&= Q_I \log \prod_{\mathbf{n} \in \mathcal{S}_3} \frac{Q_I + Q_{\mathbf{n}}}{Q_I + \Delta Q_{\mathbf{n}}} + Q_I \log \left(1 + \frac{Q_{\mathbf{n}_\circ}}{Q_I} \right) - \Delta Q_I \log \left(1 + \frac{\Delta Q_{\mathbf{n}_\circ}}{\Delta Q_I} \right),
\end{aligned}$$

where Δl_I represents the portion of the average rate unchanged by the combination of $\mathcal{G}_I$ and $\mathcal{G}_J$. It follows that $\Delta\bar{l}_I \geq 0$ since $\log \prod_{\mathbf{n} \in \mathcal{S}_3} (Q_I + Q_{\mathbf{n}})/(Q_I + \Delta Q_{\mathbf{n}}) \geq 0$ and for any constant $c > 0$, $x \log(1 + c/x)$ is monotonically increasing in $x > 0$, giving

$$\Delta Q_I \log \left(1 + \frac{\Delta Q_{\mathbf{n}_\circ}}{\Delta Q_I} \right) \leq \Delta Q_I \log \left(1 + \frac{Q_{\mathbf{n}_\circ}}{\Delta Q_I} \right) \leq Q_I \log \left(1 + \frac{Q_{\mathbf{n}_\circ}}{Q_I} \right).$$

Similarly, using Δl_J as the portion of $\bar{l}_J$ unchanged by the combination,

$$\begin{aligned}
-\bar{l}_J &= Q_J \log Q_J + \sum_{\mathbf{n}k \in \mathcal{S}_1 \cup \mathcal{S}_2} Q_{\mathbf{n}k} \log \frac{Q_{\mathbf{n}k}}{\Delta Q_{\mathbf{n}}} + \sum_{k=1}^{K(\mathbf{n}_\circ)} Q_{\mathbf{n}_\circ k} \log \frac{Q_{\mathbf{n}_\circ k}}{\Delta Q_{\mathbf{n}_\circ}} + \Delta l_J \\
-\bar{l}_J^\star &= Q_J \log(Q_J + Q_I) + \sum_{\mathbf{n}k \in \mathcal{S}_1} Q_{\mathbf{n}k} \log \frac{Q_{\mathbf{n}k} + Q_I}{\Delta Q_{\mathbf{n}} + Q_I} + \sum_{\mathbf{n}k \in \mathcal{S}_2} Q_{\mathbf{n}k} \log \frac{Q_{\mathbf{n}k}}{\Delta Q_{\mathbf{n}} + Q_I} \\
&\quad + \sum_{k=1}^{K(\mathbf{n}_\circ)} Q_{\mathbf{n}_\circ k} \log \frac{Q_{\mathbf{n}_\circ k}}{\Delta Q_{\mathbf{n}_\circ} + \Delta Q_I} + \Delta l_J
\end{aligned}$$

$$\begin{aligned}
\Delta\bar{l}_J &= Q_J \log\left(\frac{Q_J+Q_I}{Q_J}\right) + \sum_{\mathbf{n}k\in\mathcal{S}_1} Q_{\mathbf{n}k}\log\frac{Q_{\mathbf{n}k}+Q_I}{Q_{\mathbf{n}k}} + \sum_{\mathbf{n}k\in\mathcal{S}_1} Q_{\mathbf{n}k}\log\frac{\Delta Q_{\mathbf{n}}}{\Delta Q_{\mathbf{n}}+Q_I} \\
&\quad + \sum_{\mathbf{n}k\in\mathcal{S}_2} Q_{\mathbf{n}k}\log\frac{\Delta Q_{\mathbf{n}}}{\Delta Q_{\mathbf{n}}+Q_I} + \sum_{k=1}^{K(\mathbf{n}_\circ)} Q_{\mathbf{n}_\circ k}\log\frac{\Delta Q_{\mathbf{n}_\circ}}{\Delta Q_{\mathbf{n}_\circ}+\Delta Q_I} \\
&= Q_J\log\left(1+\frac{Q_I}{Q_J}\right) + \sum_{\mathbf{n}\in\mathcal{S}_1} Q_{\mathbf{n}}\log\left(1+\frac{Q_I}{Q_{\mathbf{n}}}\right) - \sum_{\mathbf{n}\in\mathcal{S}_3}\Delta Q_{\mathbf{n}}\log\left(1+\frac{Q_I}{\Delta Q_{\mathbf{n}}}\right) \\
&\quad -\Delta Q_{\mathbf{n}_\circ}\log\left(1+\frac{\Delta Q_I}{\Delta Q_{\mathbf{n}_\circ}}\right) \\
&\geq \sum_{\mathbf{n}\in\mathcal{S}_1\cup\mathcal{S}_3}\left[Q_{\mathbf{n}}\log\left(1+\frac{Q_I}{Q_{\mathbf{n}}}\right) - \Delta Q_{\mathbf{n}}\log\left(1+\frac{Q_I}{\Delta Q_{\mathbf{n}}}\right)\right].
\end{aligned}$$

Thus $\Delta\bar{l}_J \geq 0$ by the monotonicity of $x\log(1+c/x)$. Since the optimal rates of $\mathcal{G}_I$ and $\mathcal{G}_J$ both decrease after combining, I have the desired result. □

Unfortunately, Theorem 5 does not hold for matched Huffman coding. For the example given in Figure 3.5, if the p.m.f. on $\mathcal{X} = \{a_0, a_1, \ldots, a_7\}$ is $\{0.45, 0.1, 0.05, 0.1, 0.1, 0.1, 0.05, 0.05\}$, then the matched Huffman code rates for the partition before and after combining are 2.25 and 2.3 respectively (shown in Figures 3.5(c) and (d)). Theorem 6 shows a weaker result that does apply in Huffman coding.

Theorem 6 *Given partition $\mathcal{P}(\mathcal{X})$ of $\mathcal{X}$ on $p(x,y)$, if $\mathcal{G}_I, \mathcal{G}_J \in \mathcal{P}(\mathcal{X})$ satisfy: (1) $\mathcal{G}_I$ is a 1-level group and (2) $\mathcal{G}_I$ can be combined with $\mathcal{G}_J$ at root J of $\mathcal{T}(\mathcal{G}_J)$ to form partition $\mathcal{P}^\star(\mathcal{X})$, then $El^{(H)}_{\mathcal{P}^\star(\mathcal{X})}(X) \leq El^{(H)}_{\mathcal{P}(\mathcal{X})}(X)$.*

Proof: Let γ_X denote the matched Huffman code for $\mathcal{P}(\mathcal{X})$, and let α_I and α_J be this code's binary descriptions for nodes I and J. The binary description for any symbol in $\mathcal{G}_I$ equals α_I ($\gamma_X(x) = \alpha_I$ for each $x \in \mathcal{G}_I$) while the binary description for any symbol in $\mathcal{G}_J$ has prefix α_J ($\gamma_X(x) = \alpha_J\gamma'_X(x)$ for each $x \in \mathcal{G}_J$, where γ'_X is a matched Huffman code for $\mathcal{G}_J$). Let $\alpha_{\min}$ be the shorter of α_I and α_J. Since γ_X is a matched Huffman code for $\mathcal{P}(\mathcal{X})$ and $\mathcal{P}^\star(\mathcal{X})$

is a partition of $\mathcal{X}$ on $p(x,y)$,

$$\gamma_X^\star(x) = \begin{cases} \alpha_{\min} & \text{if } x \in \mathcal{G}_I \\ \alpha_{\min}\gamma'_X(x) & \text{if } x \in \mathcal{G}_J \\ \gamma_X(x) & \text{otherwise} \end{cases}$$

is a matched code for $\mathcal{P}^\star(\mathcal{X})$. Further, $|\alpha_{\min}| \leq |\alpha_I|$ and $|\alpha_{\min}| \leq |\alpha_J|$ imply that the expected length of $\alpha^\star(X)$ is less than or equal to the expected length of $\gamma_X(X)$ (but perhaps greater than the expected length of the matched Huffman code for $\mathcal{P}^\star(\mathcal{X})$). □

3.2.6 Partition Design and Complexity

I next consider techniques for finding the optimal partition for a fixed $p(x,y)$. These techniques may be used to find the optimal partition for Huffman coding or the optimal partition for arithmetic coding by applying the appropriate optimality criterion. Since optimal lossless SISC design is NP-hard [32] and matched code design requires only polynomial complexity, the partition design problem must be NP-hard. I next describe a search algorithm for the optimal solution. The approach given gains efficiency by taking advantage of the above-described properties of optimal partitions.

I build an optimal partition for $\mathcal{X}$ by building optimal groups for larger and larger subsets $\mathcal{X}' \subseteq \mathcal{X}$ for which $\mathcal{R}(\mathcal{X}') = \{x \in \mathcal{X}' : \{x\} \neq (\mathcal{X}' \cap \{x\}^c)\}$ implies $\mathcal{R}(\mathcal{X}') \neq \phi$ and testing all legitimate combinations of those groups. I eliminate all $\mathcal{X}'$ with $\mathcal{R}(\mathcal{X}') = \phi$ by Lemma 2. By Lemma 3, the optimal group for $\mathcal{X}'$ is $\mathcal{G}^\star(\mathcal{X}') = (\mathcal{R}(\mathcal{X}') : \mathcal{C}(\mathcal{X}'))$, where $\mathcal{C}(\mathcal{X}') = \mathcal{P}^\star(\mathcal{X}' \cap \mathcal{R}(\mathcal{X}')^c)$ is the optimal partition on $\mathcal{X}' \cap \mathcal{R}(\mathcal{X}')^c$. Thus $\mathcal{G}^\star(\{x\}) = (x)$ for any $x \in \mathcal{X}$. For any $\mathcal{X}' \subseteq \mathcal{X}$ with $|\mathcal{R}(\mathcal{X}')| > 0$ and $|\mathcal{X}' \cap \mathcal{R}(\mathcal{X}')^c| > 0$, I find $\mathcal{C}(\mathcal{X}')$ by

calculating the expected rate of the matched code for each set of groups of the form

$$\mathcal{C} = \{\mathcal{G}^\star(\mathcal{S}_1), \ldots, \mathcal{G}^\star(\mathcal{S}_K) : \quad |\mathcal{R}(\mathcal{S}_k)| > 0 \ \ \forall k, \ \ \cup_{k=1}^{K} \mathcal{S}_k = (\mathcal{X}' \cap \mathcal{R}(\mathcal{X}')^c), \mathcal{S}_i \cap \mathcal{S}_j = \phi,$$
$$\mathcal{S}_i, \mathcal{S}_j \text{ can't be combined by Theorem 5 or 6 } \forall i, j\},$$

and choosing the one with the best performance. By examining all subsets of $\mathcal{X}$ in order of increasing size and storing the results, all $\mathcal{G}^\star(\mathcal{S})$ values are available when needed and calculation repetition can be avoided. (For example, $\mathcal{X}' \neq \mathcal{X}''$ does not imply that $(\mathcal{X}' \cap \mathcal{R}(\mathcal{X}')^c) \neq (\mathcal{X}'' \cap \mathcal{R}(\mathcal{X}'')^c)$, but we can avoid repeating the optimization through appropriate storage of past results.)

I consider both design complexity and operation complexity for the encoder and decoder.

The design complexity varies greatly depending $p(x)$ and the locations of the zeros in $p(x, y)$ [24]. The worst case complexity can be loosely bounded from above by $\sum_{k=1}^{|\mathcal{X}|} \binom{|\mathcal{X}|}{k} B_k < 2^{|\mathcal{X}|} B_{|\mathcal{X}|}$, where B_m is the Bell number. The Bell number $B_m \sim m^{-1/2}[\lambda(m)]^{m+1/2} e^{\lambda(m)-m-1}$ is the number of ways a set of m elements can be partitioned into nonempty subsets, where $\lambda(m) \ln[\lambda(m)] = m$ [34]. When the number of symbols that are not confusable is low or high the actual complexity is greatly reduced (e.g., the complexity is lowest when no symbols can be combined or all symbols can be combined).

While optimal SISC design is expensive, the encoding and decoding complexities for an optimal SISC are comparable to the encoding and decoding complexities of a traditional (single-sender, single-receiver) Huffman or arithmetic code. All are linear in $|\mathcal{X}|$. For Huffman coding, I use a table look-up encoder and a binary tree decoder. The decoder's binary tree labels node $\mathbf{n}$ with all $x \in \mathcal{X}$ such that $\gamma_X(x) = \mathbf{n}$. Since the decoder knows y, it stops reading bits when it reaches a node $\mathbf{n}$ for which there is some $x \in \mathbf{n}$ with $p(x, y) > 0$; the

decoder outputs that x. Similarly, an arithmetic SISC has encoding and decoding complexities that are comparable to those of traditional arithmetic codes. In arithmetic coding, the encoder describes x by describing the interval for $\mathbf{n}$ such that $x \in \mathbf{n}$, and the decoder maps that interval back to the $x \in \mathbf{n}$ with $p(x, y) > 0$.

3.3 Lossless Instantaneous Multiple Access Source Codes

3.3.1 Problem Statement, Partition Pairs, and Optimal Matched Codes

I here drop the SISC assumption that Y (or X) can be decoded independently and consider MASC design when it may be necessary to decode the two symbol descriptions together. I replace the SISC partition $\mathcal{P}(\mathcal{X})$ by a pair of partitions $(\mathcal{P}(\mathcal{X}), \mathcal{P}(\mathcal{Y}))$ that describe the prefix and equivalence relationships for $\{\gamma_X(x) : x \in \mathcal{X}\}$ and $\{\gamma_Y(y) : y \in \mathcal{Y}\}$, respectively.

For an MASC to be *instantaneous*, the decoder must recognize when it reaches the end of $\gamma_X(X)$ and $\gamma_Y(Y)$. We again use tree structures to help us understand the prefix relationships that make instantaneous decoding possible. Let $\mathcal{T}_\mathcal{X}$ and $\mathcal{T}_\mathcal{Y}$ be a pair of binary trees for which each symbol $x \in \mathcal{X}$ resides at the node reached by traversing path $\gamma_X(x)$ from the root of $\mathcal{T}_\mathcal{X}$ and each symbol $y \in \mathcal{Y}$ resides at the node reached by traversing path $\gamma_Y(y)$ from the root of $\mathcal{T}_\mathcal{Y}$. To decode binary strings $\gamma_X(X)\ldots$ and $\gamma_Y(Y)\ldots$, the decoder starts at the roots of $\mathcal{T}_\mathcal{X}$ and $\mathcal{T}_\mathcal{Y}$ and moves down the first few bits of the path $\gamma_X(X)\ldots$ in $\mathcal{T}_\mathcal{X}$ and $\gamma_Y(Y)\ldots$ in $\mathcal{T}_\mathcal{Y}$, in each case stopping when it reaches an occupied node. Let $\mathbf{n}_X$ and $\mathbf{n}_Y$ denote those occupied nodes, and use $\mathcal{T}_X$ and $\mathcal{T}_Y$ to describe the subtrees comprising, respectively, $\mathbf{n}_X$

plus all of its descendants and $\mathbf{n}_Y$ plus all of its descendants. For instantaneous coding, at least one of the following conditions must hold:

(A) $X \in \mathcal{T}_X$ or $\mathbf{n}_Y$ is a leaf implies that $Y \in \mathbf{n}_Y$, and $Y \in \mathcal{T}_Y$ or $\mathbf{n}_X$ is a leaf implies that $X \in \mathbf{n}_X$;

(B) $X \in \mathcal{T}_X$ implies that $Y \notin \mathbf{n}_Y$;

(C) $Y \in \mathcal{T}_Y$ implies that $X \notin \mathbf{n}_X$.

Under condition (A), the decoder has reached the end of $\gamma_X(X)$ and $\gamma_Y(Y)$. Under condition (B), the decoder reads the next few bits of $\gamma_Y(Y)\ldots$, traversing the described path in $\mathcal{T}_Y$ to node $\mathbf{n}'_Y$ with subtree $\mathcal{T}'_Y$. Condition (C) similarly leads to a new node $\mathbf{n}'_X$ and subtree $\mathcal{T}'_X$. If none of these conditions holds, then the decoder is not instantaneous since it cannot determine whether to continue reading one or both of the descriptions. The decoder continues the above procedure until it determines the nodes of $\mathcal{T}_\mathcal{X}$ and $\mathcal{T}_\mathcal{Y}$ where X and Y reside. At each step before the decoding halts, at least one of the conditions (A), (B), and (C) must be satisfied.

For an MASC to be *lossless*, the above procedure's final nodes $\mathbf{n}_X$ and $\mathbf{n}_Y$ must satisfy $(X, Y) \in \mathbf{n}_X \times \mathbf{n}_Y$, and for any other $(x', y') \in \mathbf{n}_X \times \mathbf{n}_Y$, we must have $p(X, y') = p(x', Y) = p(x', y') = 0$.

I define a partition pair $(\mathcal{P}(\mathcal{X}), \mathcal{P}(\mathcal{Y}))$ to be any pair of prefix relationships on $\{\gamma_X(x) : x \in \mathcal{X}\}$ and $\{\gamma_Y(y) : y \in \mathcal{Y}\}$. The following theorem gives a simple test for determining whether $(\mathcal{P}(\mathcal{X}), \mathcal{P}(\mathcal{Y}))$ gives a lossless instantaneous MASC. Theorem 7 reduces to Lemma 1 when either $\mathcal{P}(\mathcal{X}) = \{(x) : x \in \mathcal{X}\}$ or $\mathcal{P}(\mathcal{Y}) = \{(y) : y \in \mathcal{Y}\}$. In either of these cases, the general MASC problem reduces to the SISC problem of Section 3.2.

Theorem 7 *(MASC Prefix Property): Partition pair* $(\mathcal{P}(\mathcal{X}), \mathcal{P}(\mathcal{Y}))$ *yields a lossless instantaneous MASC for* $p(x,y)$ *if and only if for any* $(x,y) \neq (x',y')$ *with* $p(x,y) > 0$ *and* $p(x',y') > 0$*, at least one of* $\{\gamma_X(x), \gamma_X(x')\}$ *and* $\{\gamma_Y(y), \gamma_Y(y')\}$ *is prefix free.*

Proof: If an MASC is not instantaneous, then there must be a time in the decoding procedure when the decoder reaches nodes $(\mathbf{n}_X, \mathbf{n}_Y)$ with subtrees $(\mathcal{T}_X, \mathcal{T}_Y)$ but none of conditions (A), (B), or (C) is satisfied. Violating (A), (B), and (C) implies that there must exist a pair $(x,y),(x',y') \in \mathcal{T}_X \times \mathcal{T}_Y$ with $p(x,y) > 0$ and $p(x',y') > 0$ such that either $(x,x') \in \mathbf{n}_X \times (\mathcal{T}_X \cap \mathbf{n}_X^c)$ and $(y,y') \in \mathbf{n}_Y \times \mathbf{n}_Y$ or $(x,x') \in \mathbf{n}_X \times (\mathcal{T}_X \cap \mathbf{n}_X^c)$ and $(y,y') \in \mathbf{n}_Y \times (\mathcal{T}_Y \cap \mathbf{n}_Y^c)$ or $(x,x') \in \mathbf{n}_X \times \mathbf{n}_X$ and $(y,y') \in \mathbf{n}_Y \times (\mathcal{T}_Y \cap \mathbf{n}_Y^c)$. Thus both $\{\gamma_X(x), \gamma_X(x')\}$ and $\{\gamma_Y(y), \gamma_Y(y')\}$ violate the prefix property. If an MASC is instantaneous but not lossless, then there must be a pair of nodes $(\mathbf{n}_X, \mathbf{n}_Y)$ and an $(x,y) \neq (x',y')$ for which $(x,y),(x',y') \in \mathbf{n}_X \times \mathbf{n}_Y$ and both $p(x,y) > 0$ and $p(x',y') > 0$, so that the decoder reaches a node pair instantaneously but cannot decode without loss. In this case, at least one of the following must be true: (1) $x \neq x'$ and $\gamma_X(x) = \gamma_X(x')$ or (2) $y \neq y'$ and $\gamma_X(y) = \gamma_X(y')$. In either case, the MASC prefix condition is violated.

I now show that if the MASC prefix condition is violated, then we cannot achieve a lossless instantaneous MASC. I begin by building two binary trees, $\mathcal{T}_\mathcal{X}$ and $\mathcal{T}_\mathcal{Y}$. For each $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, I place symbol x at the node reached by path $\gamma_X(x)$ in $\mathcal{T}_\mathcal{X}$ and symbol y at the node reached by path $\gamma_Y(y)$ in $\mathcal{T}_\mathcal{Y}$. If there exists a violation of the MASC prefix condition, then there exists an $(x,y) \neq (x',y')$ for which $p(x,y) > 0$ and $p(x',y') > 0$ and neither $\{\gamma_X(x), \gamma_X(x')\}$ nor $\{\gamma_Y(y), \gamma_Y(y')\}$ is prefix free. If $\mathbf{n}_x$, $\mathbf{n}_{x'}$, $\mathbf{n}_y$, and $\mathbf{n}_{y'}$ are the nodes in our tree construction satisfying $x \in \mathbf{n}_x$, $x' \in \mathbf{n}_{x'}$, $y \in \mathbf{n}_y$, and $y' \in \mathbf{n}_{y'}$, then one of two

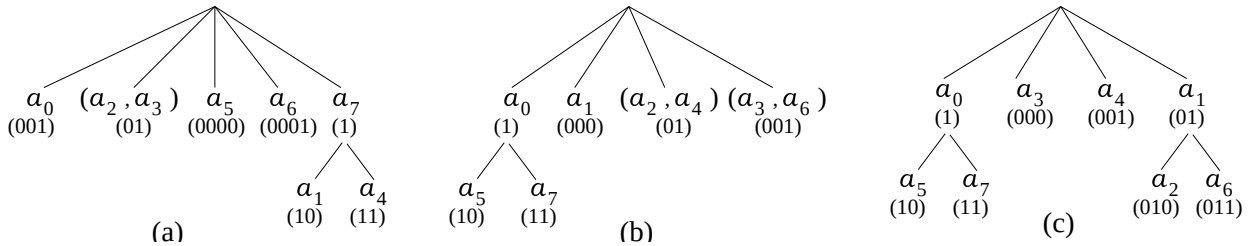


Figure 3.7: The partition pair $(\mathcal{P}(\mathcal{X}), \mathcal{P}(\mathcal{Y}))$ with $\mathcal{P}(\mathcal{X})$ shown in (a) and $\mathcal{P}(\mathcal{Y})$ shown in (b) gives a lossless, instantaneous MASC for the p.m.f. in Table 6.1(a). Replacing $\mathcal{P}(\mathcal{Y})$ with the partition shown in (c) fails to give a lossless, instantaneous MASC for the same p.m.f.

cases can occur. If $(\mathbf{n}_x, \mathbf{n}_y) = (\mathbf{n}_{x'}, \mathbf{n}_{y'})$, then the example satisfies (A); in this case, the code is not lossless since $\gamma_X(x) = \gamma_X(x')$ and $\gamma_Y(y) = \gamma_Y(y')$ (by construction) and $p(x, y) > 0$ and $p(x', y') > 0$ (by assumption). If $(\mathbf{n}_x, \mathbf{n}_y) \neq (\mathbf{n}_{x'}, \mathbf{n}_{y'})$, then either one of $\{\mathbf{n}_x, \mathbf{n}_{x'}\}$ is the ancestor of the other and $\{\mathbf{n}_y, \mathbf{n}_{y'}\}$ is not prefix free or one of $\{\mathbf{n}_y, \mathbf{n}_{y'}\}$ is the ancestor of the other and $\{\mathbf{n}_x, \mathbf{n}_{x'}\}$ is not prefix free (or both). In this case, none of (A), (B), and (C) is satisfied since the decoder cannot determine whether or not to read beyond the common prefix of $\{\gamma_X(x), \gamma_X(x')\}$ in the description of X or the common prefix of $\{\gamma_Y(y), \gamma_Y(y')\}$ in the description of Y. □

Example: Again consider the p.m.f. in Table 6.1(a). If we set $\mathcal{P}(\mathcal{X})$ and $\mathcal{P}(\mathcal{Y})$ to be the partitions in Figures 3.7(a) and (b), respectively, then there is no pair $(x, y) \neq (x', y')$ with $p(x, y) > 0$ and $p(x', y') > 0$ for which neither $\{\gamma_X(x), \gamma_X(x')\}$ nor $\{\gamma_Y(y), \gamma_Y(y')\}$ is prefix free. Thus $(\mathcal{P}(\mathcal{X}), \mathcal{P}(\mathcal{Y}))$ gives a lossless, instantaneous code for the p.m.f. in Table 6.1(a). In contrast, if $\mathcal{P}(\mathcal{X})$ and $\mathcal{P}(\mathcal{Y})$ are the partitions in Figures 3.7(a) and (c), respectively, then $p(a_2, a_2) > 0$ and $p(a_3, a_1) > 0$ but $\gamma_X(a_2) = \gamma_X(a_3)$ and $\gamma_Y(a_1) \prec \gamma_Y(a_2)$. Thus neither $\{\gamma_X(a_2), \gamma_X(a_3)\}$ nor $\{\gamma_Y(a_1), \gamma_Y(a_2)\}$ is prefix free, and the decoder cannot know whether

or not to continue reading beyond $\gamma_Y(a_1)$ in decoding the description of Y when it receives $\gamma_X(a_2) = \gamma_X(a_3)$ as its description from X.

Theorem 2 generalizes to show that every lossless, instantaneous MASC is a pair of matched codes for some $(\mathcal{P}(\mathcal{X}), \mathcal{P}(\mathcal{Y}))$ satisfying Theorem 7. Thus optimal MASC design is equivalent to optimal partition design followed by optimal matched code design. Matched code design for each partition of an MASC is identical to matched code design for the partition of an SISC. Thus the generalization to optimal matched Huffman and arithmetic codes for any partition pair $(\mathcal{P}(\mathcal{X}), \mathcal{P}(\mathcal{Y}))$ for $p(x, y)$ is immediate. The codewords of an optimal matched Huffman code for the partitions in Figure 3.7 appear in parentheses under the nodes of the partition trees.

3.3.2 Optimal Partition Properties

Given a partition pair $(\mathcal{P}(\mathcal{X}), \mathcal{P}(\mathcal{Y}))$ that satisfies the MASC prefix condition, $(\mathcal{P}(\mathcal{X}), \mathcal{P}(\mathcal{Y}))$ is *optimal for use in a matched Huffman MASC* on $p(x, y)$ if $(El^{(H)}_{\mathcal{P}(\mathcal{X})}(X), El^{(H)}_{\mathcal{P}(\mathcal{Y})}(Y))$ sits on the lower boundary of the rates achievable by a lossless MASC on alphabet $\mathcal{X} \times \mathcal{Y}$. Similarly, $(\mathcal{P}(\mathcal{X}), \mathcal{P}(\mathcal{Y}))$ is *optimal for use in a matched arithmetic MASC* on $p(x, y)$ if $(El^{\star}_{\mathcal{P}(\mathcal{X})}(X), El^{\star}_{\mathcal{P}(\mathcal{Y})}(Y))$ sits on the lower boundary of

$$\{(El^{\star}_{\mathcal{P}'(\mathcal{X})}(X), El^{\star}_{\mathcal{P}'(\mathcal{Y})}(Y)) : (\mathcal{P}'(\mathcal{X}), \mathcal{P}'(\mathcal{Y})) \text{ are partitions on } \mathcal{X} \times \mathcal{Y} \text{ for } \{p(x, y)\}\}.$$

Again $l^{(H)}_{\mathcal{P}}$ and $l^{\star}_{\mathcal{P}}$ denote the Huffman and optimal description lengths respectively for partition $\mathcal{P}$, and Huffman coding is optimal over all codes on a fixed alphabet. (Mixed codes (e.g., Huffman coding on X and arithmetic coding on Y) are also possible within this framework.)

Lemma 5 *For each partition pair $(\mathcal{P}(\mathcal{X}), \mathcal{P}(\mathcal{Y}))$ that achieves performance on the lower boundary of the achievable rate region, there exists a partition pair $(\mathcal{P}^\star(\mathcal{X}), \mathcal{P}^\star(\mathcal{Y}))$ achieving the same rate performance as $(\mathcal{P}(\mathcal{X}), \mathcal{P}(\mathcal{Y}))$ for which every node except for the roots of $\mathcal{P}^\star(\mathcal{X})$ and $\mathcal{P}^\star(\mathcal{Y})$ is non-empty and no node except for the roots can have exactly one child.*

Proof: We build $\mathcal{P}^\star(\mathcal{X})$ and $\mathcal{P}^\star(\mathcal{Y})$ by modifying $\mathcal{P}(\mathcal{X})$ and $\mathcal{P}(\mathcal{Y})$ to remove all empty nodes, attaching the node's children directly to its parent. We also remove any non-root node $\mathbf{n}$ that has exactly one child $\mathbf{n}1$, combining $\mathbf{n}$ and $\mathbf{n}1$ to form 1-level group $(\mathbf{n}, \mathbf{n}1)$ with $\{\mathbf{n}1k\}_{k=1}^{K(\mathbf{n}1)}$ descending directly from $(\mathbf{n}, \mathbf{n}1)$. Since neither change can increase the code's rate or change the sets of symbols whose descriptions violated the prefix property, we have the desired $(\mathcal{P}^\star(\mathcal{X}), \mathcal{P}^\star(\mathcal{Y}))$ by Theorem 7. □

3.3.3 Partition Design and Complexity

For a fixed partition $\mathcal{P}(\mathcal{Y})$ on $\mathcal{Y}$ with matched code γ_Y, I design the optimal partition and matched code γ_X on $\mathcal{X}$ such that (γ_X, γ_Y) satisfy the MASC prefix condition. Traversing through all partitions on $\mathcal{Y}$, I can trace out the lower boundary of achievable rates for MASC.

A very loose bound on the worst-case complexity for designing an optimal MASC is $C_{\text{MASC}} < 2^{|\mathcal{X}|+|\mathcal{Y}|} B_{|\mathcal{X}|}(|\mathcal{Y}|+1)!$. The encoding and decoding complexities of the proposed optimal MASCs are linear in the alphabet size and comparable to the corresponding traditional codes.

3.4 Near-Lossless Instantaneous Multiple Access Source Codes

Finally, I generalize from lossless to near-lossless codes. For any fixed small $\epsilon > 0$, I call MASC $((\gamma_X, \gamma_Y), \gamma^{-1})$ a *near-lossless instantaneous MASC for* $P_e \leq \epsilon$ if $((\gamma_X, \gamma_Y), \gamma^{-1})$ yields instantaneous decoding with $P_e = \Pr(\gamma^{-1}(\gamma_X(X), \gamma_Y(Y)) \neq (X, Y)) \leq \epsilon$. Since the code is instantaneous, the decoder builds its reconstruction $(\hat{x}_1, \hat{y}_1)$ of (x_1, y_1) using exactly $|\gamma_X(x_1)|$ bits from $\gamma_X(x_1)\gamma_X(x_2)\gamma_X(x_3)\ldots$ and $|\gamma_Y(y_1)|$ bits from $\gamma_Y(y_1)\gamma_Y(y_2)\gamma_Y(y_3)\ldots$ (without prior knowledge of these lengths). Thus even when $(\hat{x}_1, \hat{y}_1) \neq (x_1, y_1)$, the decoder correctly determines $|\gamma_X(x_1)|$ and $|\gamma_Y(y_1)|$. This requirement disallows loss of synchronization and error propagation.

Theorem 8 *(Near-Lossless MASC Prefix Property): Partition pair* $(\mathcal{P}(\mathcal{X}), \mathcal{P}(\mathcal{Y}))$ *yields a near-lossless instantaneous MASC for* $p(x, y)$ *if and only if for any* $(x, y) \neq (x', y')$ *with* $p(x, y) > 0$ *and* $p(x', y') > 0$, *either:*

(A) at least one of $\{\gamma_X(x), \gamma_X(x')\}$ *and* $\{\gamma_Y(y), \gamma_Y(y')\}$ *is prefix free; or*

(B) $\gamma_X(x) = \gamma_X(x')$ *and* $\gamma_Y(y) = \gamma_Y(y')$.

Proof: Recall that $\gamma_X(x)$ is a *proper* prefix of $\gamma_X(x')$ (written $\gamma_X(x) \prec \gamma_X(x')$) if $\gamma_X(x) \preceq \gamma_X(x')$ and $\gamma_X(x) \neq \gamma_X(x')$. Fix some $(x, y) \neq (x', y')$ with $p(x, y) > 0$ and $p(x', y') > 0$. Under (A), the decoder can instantaneously and losslessly distinguish between (x, y) and (x', y') by Theorem 7. Under (B), we cannot decode losslessly, but there is no ambiguity in how many bits to decode.

If neither (A) nor (B) is satisfied, then there exists an $(x, y) \neq (x', y')$ for which $p(x, y) > 0$ and $p(x', y') > 0$ and either the decoder cannot determine whether to decode $|\gamma_X(x)|$ bits or $|\gamma_X(x')| > |\gamma_X(x)|$ bits because $\gamma_X(x) \prec \gamma_X(x')$ and $\{\gamma_Y(y), \gamma_Y(y')\}$ is not prefix free or the decoder cannot determine whether to decode $|\gamma_Y(y)|$ bits or $|\gamma_Y(y')| > |\gamma_Y(y)|$ bits because $\gamma_Y(y) \prec \gamma_Y(y')$ and $\{\gamma_X(x), \gamma_X(x')\}$ is not prefix free. □

In a near-lossless SISC for X given Y, the prefix condition simplifies to: for any $x, x' \in \mathcal{X}$ for which there exists a $y \in \mathcal{Y}$ with $p(x, y) > 0$ and $p(x', y) > 0$, $\gamma_X(x) \prec \gamma_X(x')$ is disallowed (as in lossless coding) but $\gamma_X(x) = \gamma_X(x')$ is allowed. Here $\gamma_X(x) \prec \gamma_X(x')$ would leave the decoder no means of determining whether to decode $|\gamma_X(x)|$ bits or $|\gamma_X(x')|$ bits. However, $\gamma_X(x) = \gamma_X(x')$ allows instantaneous (but not error free) decoding.

Given a partition pair $(\mathcal{P}(\mathcal{X}), \mathcal{P}(\mathcal{Y}))$ satisfying the near-lossless MASC prefix property, then for any $x \in \mathbf{n}_x \in \mathcal{T}(\mathcal{P}(\mathcal{X}))$ and $y \in \mathbf{n}_y \in \mathcal{T}(\mathcal{P}(\mathcal{Y}))$ with $p(x, y) > 0$, the optimal decoder gives

$$\begin{aligned} \gamma^{-1}(\gamma_X(x), \gamma_Y(y)) &= \arg \max_{(x', y') \in \mathbf{n}_x \times \mathbf{n}_y} p(x', y') \\ P_e(\mathcal{P}(\mathcal{X}), \mathcal{P}(\mathcal{Y})) &= \sum_{(\mathbf{n}_\mathcal{X}, \mathbf{n}_\mathcal{Y}) \in \mathcal{T}(\mathcal{P}(\mathcal{X})) \times \mathcal{T}(\mathcal{P}(\mathcal{Y}))} \left[\sum_{(x,y) \in \mathbf{n}_\mathcal{X} \times \mathbf{n}_\mathcal{Y}} p(x, y) - \max_{(x,y) \in \mathbf{n}_\mathcal{X} \times \mathbf{n}_\mathcal{Y}} p(x, y) \right]. \end{aligned}$$

By Theorem 8, I can design a near-lossless MASC by designing a lossless MASC on a reduced alphabet that represents each 1-level group by a single symbol. The optimal near-lossless MASC can be found by searching the reduced alphabets that satisfy a given error constraint ϵ. The optimal performance of this one-dimensional code is bounded below by the convex hull of the Slepian-Wolf rate regions on these reduced alphabets.

It is interesting to compare the near-lossless coding approach given above to MASCs based on error correction codes (see for example [25, 26, 29, 30, 31]). The strengths of

those algorithms are that they are computationally efficient at high coding dimension n and achieve good coding performance (low error probabilities and rates close to the Slepian-Wolf bound) when the relationship between sources X and Y resembles that between the input and output of the noisy channels for which the error correction code was designed (e.g., the structured sources of [25, 26, 29, 30, 31]). The weaknesses of these codes are that they do not give instantaneous coding, and they can suffer catastrophic decoding failure due to loss of synchronization when the "errors" between X and Y exceed the code's correction capabilities. In contrast, the strengths of our codes are that they are instantaneous and cannot suffer catastrophic failures. The weaknesses are that code design complexity becomes prohibitive for large coding dimensions, and thus the codes' rate and error probabilities generally fail to meet their asymptotic limits. For example, the smallest error probability that can give a result different from lossless coding for a block-length-n code is $\min\{p^n(x^n, y^n) : p^n(x^n, y^n) > 0\}$.

3.5 Low Complexity Multiple Access Source Coding Algorithms

3.5.1 Problem Statement

Since the complexity of optimal MASC design is high, in this section, I introduce a family of low complexity code design algorithms that approximates the optimal solution for instantaneous lossless and near-lossless SISCs. The algorithms may be used to design both Huffman and arithmetic SISCs for an arbitrary probability mass function $p(x, y)$. The results apply

to MASC design under the assumption of a fixed, known $\mathcal{P}(\mathcal{Y})$. The only difference between SISC and MASC design lies in the prefix property and group definitions.

Section 3.5.2 treats Alon and Orlitsky's elegant chromatic entropy bound on optimal SISC rates [24]; I show that the underlying code construction is NP-hard and therefore is not a viable alternative for sub-optimal code design. Section 3.5.3 introduces a constrained SISC design problem and its polynomial-time optimal solution. Section 3.5.4 describes a family of search strategies for the unconstrained problem built from the design algorithm for the constrained problem.

3.5.2 Obtaining Chromatic Entropy is NP-Hard

Since design of optimal lossless SISCs [32] and MASCs is NP-hard, I consider sub-optimal alternatives. The design used to bound the optimal SISC rate in [24] seems an attractive alternative. I next consider its complexity.

Associate with each $p(x, y)$ on $\mathcal{X} \times \mathcal{Y}$ a graph $G = (\mathcal{X}, E_{\mathcal{X}})$. Distinct vertices $x, x' \in \mathcal{X}$ are connected if and only if $p(x, y)p(x', y) > 0$ for some $y \in \mathcal{Y}$. Code γ_X is valid if and only if for every edge $(x, x') \in E_{\mathcal{X}}$, $\{\gamma_X(x), \gamma_X(x')\}$ satisfies the prefix condition; a valid code is a lossless instantaneous SISC [11].

A legitimate coloring c colors the vertices of G so that no two connected vertices have the same color. The entropy of c is $H[c(X)] = -\sum_\beta P[c^{-1}(\beta)] \log P[c^{-1}(\beta)]$, where $c^{-1}(\beta)$ describes all symbols with color β and $P[A] = \sum_{(x,y)\in A\times\mathcal{Y}} p(x, y)$. The chromatic entropy is $H_\chi(G, P) = \min H[c(X)]$ (the minimum is taken over legitimate colorings). By [24,

Theorem 2], the optimal rate R for an SISC on $p(x, y)$ satisfies

$$H_\chi(G, P) - \log[H_\chi(G, P) + 1] - \log e \leq R \leq H_\chi(G, P) + 1.$$

I prove that calculating $H_\chi(G, P)$ is NP-hard by showing that the H_χ decision problem is NP-complete. The H_χ decision problem inputs graph $G(V, E)$ and marginal pmf P and outputs the answer to "Is $H_\chi(G, P) \leq \log_2 3$?"

Theorem 9 *H_χ is NP-complete.*

Proof: Given any coloring on G as a guess, we can always verify in polynomial time if this coloring has an entropy $\leq \log_2 3$. Thus $H_\chi \in NP$.

I next show that there exists a polynomial reduction of $3C$ to H_χ. (Here $3C$ denotes the problem "is G colorable with 3 colors?", which is NP-complete [35].) The input I of $3C$ is graph $G'(V', E')$; I give a polynomial algorithm to construct input $f(I)$ of H_χ from I; then I show that G' is 3-colorable if and only if $H_\chi(G, P) \leq \log_2 3$.

Construct $G(V, E)$ as: $V = V' \cup \{v_1, v_2, v_3\} = \{v'_1, \ldots, v'_M, v_1, v_2, v_3\}$ $(M = |V'|)$, $E = E' \cup \{(v_1, v_2), (v_1, v_3), (v_2, v_3)\}$, i.e., v_1, v_2, v_3 form a triangle. Construct P as $P(v_1) = 1/3$, $P(v_2) = 1/3$, $P(v_3) = 1/3 - 1/M$, and $P(v'_j) = 1/M^2$ for $j = 1, \ldots, M$.

Assume G' is 3-colorable, then G is also 3-colorable. Thus $H_\chi(G, P) \leq \log_2 3$.

Next, assume G' is K-colorable with $K > 3$ but not 3-colorable. Then $M \geq 4$ and G is also K-colorable but not 3-colorable. Let $k_j, j = 1, 2, \ldots, K$ be the number of vertices in G' that are colored color c_j, and without loss of generality assume $k_1 \geq k_2 \geq \ldots \geq k_K$. Then a simple minimization gives:

$$H_\chi(G, P) = f\left(\frac{k_1}{M^2} + \frac{1}{3}\right) + f\left(\frac{k_2}{M^2} + \frac{1}{3}\right) + f\left(\frac{k_3}{M^2} + \frac{1}{3} - \frac{1}{M}\right) + \sum_{j=4}^{K} f\left(\frac{k_j}{M^2}\right),$$

where $f(p) = -p \log p$. The minimal value of $H_\chi(G,P)$ is achieved when the k_j's take their boundary values: $k_1 = M-(K-1)$, $k_j = 1 (2 \le j \le K)$. Thus we have

$$\min_{(G,P)\ :\ K-\text{colorable}} H_\chi(G,P)$$
$$= f\left(\frac{M-K+1}{M^2}+\frac{1}{3}\right) + f\left(\frac{1}{M^2}+\frac{1}{3}\right) + f\left(\frac{1}{M^2}+\frac{1}{3}-\frac{1}{M}\right) + (K-3) f\left(\frac{1}{M^2}\right)$$

Since $(a+b)\log(a+b) \ge a\log a + b\log b$, we have

$$\min_{K\ge 4} \min_{(G,P)\ :\ K\text{-colorable}} H_\chi(G,P) = \min_{(G,P)\ :\ 4\text{-colorable}} H_\chi(G,P)$$
$$= f\left(\frac{M-3}{M^2}+\frac{1}{3}\right) + f\left(\frac{1}{M^2}+\frac{1}{3}\right) + f\left(\frac{1}{M^2}+\frac{1}{3}-\frac{1}{M}\right) + f\left(\frac{1}{M^2}\right) \stackrel{\text{def}}{=} H_4(M)$$

It can be shown that $\frac{\mathrm{d}^3 H_4(M)}{\mathrm{d}M^3} > 0$ and $\lim_{M\to\infty} \frac{\mathrm{d}^2 H_4(M)}{\mathrm{d}M^2} = 0$, thus $\frac{\mathrm{d}^2 H_4(M)}{\mathrm{d}M^2} < 0$. Since $\frac{\mathrm{d}H_4(M)}{\mathrm{d}M}|_{M=4} < 0$, we have $\frac{\mathrm{d}H_4(M)}{\mathrm{d}M} < 0$ for $M \ge 4$. It can be verified that $\lim_{M\to\infty} H_4(M) = \log_2 3$, giving $H_4(M) > \log_2 3$ for $M > 3$.

Thus G' is 3-colorable if and only if $H_\chi(G,P) \le \log_2 3$. □

3.5.3 Constrained Side Information Source Code Design Algorithms

Given the high complexity of optimal partition design, I temporarily restrict my attention to defining and solving a constrained partition design problem.

Order alphabet $\mathcal{X}$ as $\mathcal{O} = \{x_1, \ldots, x_N\}$, where $N = |\mathcal{X}|$ and $i < j$ implies x_i precedes x_j in the chosen order. An *order-constrained partition for* $\mathcal{O}$ is any partition $\mathcal{P}$ on $\mathcal{X}$ such that a depth-first search [35] of $\mathcal{P}$ can describe $\mathcal{X}$ in order $\mathcal{O}$.

Any order-constrained partition has the property that

1. any one-level group $\mathbf{n} \in \mathcal{T}(\mathcal{P})$ is a sequence of adjacent elements in the ordering;

2. if we define the position of a group to span from its first member to its last, then the root and descendants of every multi-level group $\mathcal{T}(\mathcal{G}) \in \mathcal{T}(\mathcal{P})$ must hold adjacent positions in the ordering.

Since partitions don't specify the order of symbols in a node or siblings in the tree and since a depth-first search doesn't preserve the structure of $\mathcal{P}$, many partitions can give the same order and many orders can come from the same partition. The optimal order-constrained partition for order $\mathcal{O}$ and constant $\lambda \geq 0$ is $\mathcal{P}(\mathcal{O}, \lambda) = \arg\min_{\mathcal{P}} J_\lambda(\mathcal{P})$, where $J_\lambda(\mathcal{P}) = R(\mathcal{P}) + \lambda P_e(\mathcal{P})$ and the minimum is taken over partitions with order $\mathcal{O}$. The λ-optimal order is $\mathcal{O}^\star(\lambda) = \arg\min_{\mathcal{O}} J_\lambda(\mathcal{P}(\mathcal{O}, \lambda))$. I control P_e by varying λ.

I focus on near-lossless SISC design in the following descriptions. Since lossless and near-lossless SISC design differ only in the prefix property, extension to lossless SISC design is straightforward.

Theorem 10 *The worst-case complexity in constructing the optimal order-constrained partition and matched code for a given order* $\{x_1, \ldots, x_N\}$ *is* $O(N^4)$.

Proof: Fix λ, and let $\mathcal{G}[i,k] = \mathcal{P}(\{x_i, \ldots, x_k\}, \lambda)$, $r[i,k] = R(\mathcal{G}[i,k])$, and $e[i,k] = P_e(\mathcal{G}[i,k])$. I use dynamic programming to find $\mathcal{G}[i,k]$ for all $1 \leq k-i \leq N-i$. (Note that $\mathcal{G}[i,i] = (x_i)$ and $r[i,i] = e[i,i] = 0$ for all i.)

Let $r_m[i,j,k]$ and $e_m[i,j,k]$ denote the rate and error probability resulting from combining $\mathcal{G}[i,j]$ with $\mathcal{G}[j+1,k]$ into $\mathcal{G}[i,k]$ using a type-m combination, $m = 1, 2$. When $m = 1$, $\mathcal{G}[i,k]$ comprises an empty root with children $\mathcal{G}[i,j]$ and $\mathcal{G}[j+1,k]$; thus $e_1[i,j,k] = e[i,j]+e[j+1,k]$,

and [5]

$$r_1[i,j,k] = \begin{cases} r[i,j] + r[j+1,k] + P[i,k] & \text{in Huffman coding} \\ r[i,j] + r[j+1,k] + P[i,k]H(P[i,j]/P[i,k]) & \text{in arithmetic coding,} \end{cases}$$

where $P[i,j] = \sum_{\ell=i}^{j} p_X(x_\ell)$ and $H(p) = -p\log p - (1-p)\log(1-p)$. When $m = 2$, we build $\mathcal{G}[i,k]$ by combining one of $\mathcal{G}[i,j]$ and $\mathcal{G}[j+1,k]$ with the root of the other. Let $\mathcal{R}(\mathcal{G})$ be the root of $\mathcal{G}$, $\mathcal{C}(\mathcal{G})$ be the children of $\mathcal{R}(\mathcal{G})$, and define $s[i,j,k]$ as

$$s[i,j,k] = \begin{cases} 0 & \text{if } r[i,j] = 0, r[j+1,k] = 0 \text{ and } (\mathcal{G}[i,j] \cup \mathcal{G}[j+1,k]) = \mathcal{R}(\mathcal{G}[i,k]) \\ 1 & \text{if } r[i,j] = 0, r[j+1,k] > 0 \text{ and } \mathcal{G}[i,j] \text{ can join } \mathcal{R}(\mathcal{G}[j+1,k]) \\ 2 & \text{if } r[i,j] > 0, r[j+1,k] = 0 \text{ and } \mathcal{G}[j+1,k] \text{ can join } \mathcal{R}(\mathcal{G}[i,j]) \\ 3 & \mathcal{G}[i,j] \text{ and } \mathcal{G}[j+1,k] \text{ can't be joined by type-2 combination,} \end{cases}$$

where $\mathcal{G}, \mathcal{G}'$ can be joined if the resulting partition is valid. Then

$$(r_2[i,j,k], \quad e_2[i,j,k])$$
$$= \begin{cases} (0, \quad P_e\left(\mathcal{G}[i,j] \cup \mathcal{G}[j+1,k]\right)) & \text{if } s[i,j,k] = 0 \\ (r[j+1,k], \quad P_e(\mathcal{G}[i,j] \cup \mathcal{R}(\mathcal{G}[j+1,k])) + P_e(\mathcal{C}(\mathcal{G}[j+1,k])) & \text{if } s[i,j,k] = 1 \\ (r[i,j], \quad P_e(\mathcal{R}(\mathcal{G}[i,j]) \cup \mathcal{G}[j+1,k]) + P_e(\mathcal{C}(\mathcal{G}[i,j]))) & \text{if } s[i,j,k] = 2 \\ (\infty, \quad \infty) & \text{if } s[i,j,k] = 3. \end{cases}$$

Let $m^\star = m^\star[i,j,k] = \arg\min_{m\in\{1,2\}}\{r_m[i,j,k] + \lambda e_m[i,j,k]\}$ be the optimizing method and $j^\star = j^\star[i,k] = \arg\min_{j\in\{i,i+1,\ldots,k-1\}}\{r_{m^\star}[i,j,k] + \lambda e_{m^\star}[i,j,k]\}$ the optimizing index. Then $r[i,k] = r_{m^\star}[i,j^\star,k]$, $e[i,k] = e_{m^\star}[i,j^\star,k]$, and $\mathcal{G}[i,k]$ is the group described by the type-$m^\star$ combination of $\mathcal{G}[i,j^\star]$ and $\mathcal{G}[j^\star+1,k]$.

When the procedure is complete, $\mathcal{G}[1,N]$ is the optimal order-constrained partition on order $\{x_1,\ldots,x_N\}$; $r[1,N]$ and $e[1,N]$ are its expected rate and error probability.

[5] The derivation of $r_1[i,j,k]$ is given in the Appendix.

The number of operations required to calculate $r_m[i,j,k]$ and $e_m[i,j,k]$ dominates the complexity of this algorithm. In calculating $s[i,j,k]$, we need to check whether subsets of $\mathcal{G}[i,j]$ and $\mathcal{G}[j+1,k]$ can be joined, which requires at most $\min\{j-i+1, k-j\} \leq (k-i)/2$ operations (previous join-ability results are stored). Thus the worst-case complexity is $\sum_{i=1}^{N-1}\sum_{j=i}^{N-1}\sum_{k=j+1}^{N}(k-i)/2 = O(N^4)$. □

3.5.4 Low Complexity Design Algorithms

Since order-constrained partition optimization requires only polynomial time and optimal code design is NP-hard,, optimal order design is NP-hard. I tackle this combinatorial optimization problem using simulated annealing (SA) and iterative descent techniques. In the discussion that follows, I use $J(\mathcal{O}) = J_\lambda(\mathcal{P}(\mathcal{O},\lambda)) = R(\mathcal{P}(\mathcal{O},\lambda)) + \lambda P_e(\mathcal{P}(\mathcal{O},\lambda))$ for some fixed $\lambda > 0$.

Simulated Annealing (SA)

SA [36, 37] attempts to find an optimal solution while avoiding local optima. Applying SA to optimal order design, gives the following algorithm.

1. Initialize order $\mathcal{O}$ and temperature T.
2. While the *outer loop stopping criterion* is not satisfied, do the following.
    a) While the *inner loop stopping criterion* is not satisfied, do the following.
        i) Choose a random neighbor $\mathcal{O}'$ of $\mathcal{O}$.
        ii) If $J(\mathcal{O}') \leq J(\mathcal{O})$, set $\mathcal{O} = \mathcal{O}'$.
        iii) If $J(\mathcal{O}') > J(\mathcal{O})$, set $\mathcal{O} = \mathcal{O}'$ with probability $e^{-(J(\mathcal{O}')-J(\mathcal{O}))/T}$.

b) Set $T = \rho T$ ($0 < \rho < 1$ to reduce the temperature).

3. Return the best order.

The choices of the initial order and temperature, neighbor definitions, stopping criteria and parameter ρ all affect the speed of convergence and final solution of the SA algorithm. We choose the initial order at random and set the initial temperature T_o to make the initial distribution on orders uniform. We set $\rho \in [0.9, 0.99]$ and use the symmetric neighbor relation (switching the positions of two randomly chosen elements in $\mathcal{O}$) that achieves the best performance in our tests. We stop the inner loop if the rate has not decreased for L_{in} orders in this inner loop and stop the outer loop if the rate has not decreased for L_{out} outer loop iterations.

Descent Neighbor (DN)

Iterative descent algorithms are a degenerate special case of SA algorithms. In this case, I define an asymmetrical neighbor (*descent neighbor*) relationship that guarantees $J(\mathcal{O}') \leq J(\mathcal{O})$ for all neighbors $\mathcal{O}'$ of $\mathcal{O}$. I guarantee this property by defining the neighbors of $\mathcal{O}$ to be the orders $\mathcal{O}'$ for which $\mathcal{P}(\mathcal{O}, \lambda)$ (the optimal partition for $\mathcal{O}$) is a legitimate order-constrained partition on $\mathcal{O}'$. Given order $\mathcal{O}$, we can find its descent neighbor $\mathcal{O}'$ by switching the positions of the descendants of any internal node in $\mathcal{T}(\mathcal{P}(\mathcal{O}))$ or exchanging the position of a root with its children or permuting the positions of symbols residing at the same node. Since $\mathcal{P}(\mathcal{O})$ is also an order-constrained partition for $\mathcal{O}'$, $J(\mathcal{O}') \leq J(\mathcal{O})$.

The basic DN algorithm is as follows:

1. Choose an initial order $\mathcal{O}$ at random.

2. While no more than L_{DN} orders with identical rates have been chosen, do:

   i) Choose a random descent neighbor $\mathcal{O}'$ of $\mathcal{O}$.

   ii) If $J(\mathcal{O}') < J(\mathcal{O})$, set $\mathcal{O} = \mathcal{O}'$.

   iii) If $J(\mathcal{O}') = J(\mathcal{O})$, set $\mathcal{O} = \mathcal{O}'$ with probability p_a.

   iv) Set $p_a = \rho_a p_a$ $(0 \leq \rho_a \leq 1)$.

3. Return $J(\mathcal{O})$.

Given a complexity budget equivalent to testing C orders, we can combat local minimality problems by running the DN algorithm k times with k distinct randomly chosen initial orders. We stop the algorithm once C orders have been tested and output the best order observed over all of the experiments.

Mixing SA with DN

Various mixtures of SA with DN are also possible. For example:

1. **SA + DN**: Each time we reach step 2 b) of SA, we run DN starting from $\mathcal{O}$. (Note: In the inner loop, we stop the SA chain if the rate has not decreased for L_{SA} orders in that SA chain and stop the DN chain if the rate has not decreased for L_{DN} orders in that DN chain. We stop the outer loop if the rate has not decreased over L_{out} outer loops.)

2. **DN-merged-in-SA**: Change step i) of SA to: choose a random symmetric neighbor of $\mathcal{O}$ with probability P_s, choose a random descent neighbor of $\mathcal{O}$ with probability $1 - P_s$.

3.6 Uniquely Decodable Multiple Access Source Codes

Most prior work on the properties of optimal MASCs and SISCs focuses on instantaneous codes. In this section, I extend our interest to a wider class of codes: uniquely decodable (UD) MASCs and SISCs. A *uniquely decodable MASC* for source pair (X, Y) consists of two encoders $\gamma_X : \mathcal{X} \to \{0,1\}^\star$ and $\gamma_Y : \mathcal{Y} \to \{0,1\}^\star$ and a decoder $\gamma^{-1} : \{0,1\}^\star \times \{0,1\}^\star \to \mathcal{X} \times \mathcal{Y}$, where the decoder can uniquely and perfectly reconstruct any source sequences $x_1, x_2, x_3, \ldots$ and $y_1, y_2, y_3, \ldots$ such that $p(x_i, y_i) > 0$ for all i. In simple terms, a uniquely decodable MASC has the property that any pair of encoded strings $(\gamma_X(x^n), \gamma_Y(y^n))$ for $(x^n, y^n) \in \mathcal{X}^n \times \mathcal{Y}^n$ has only one possible pair of source sequences producing it. A *uniquely decodable SISC* assumes perfect knowledge of $y_1, y_2, y_3, \ldots$ at the decoder and can uniquely and perfectly reconstruct any source sequence $x_1, x_2, x_3, \ldots$ such that $p(x_i, y_i) > 0$ for all i. The class of UD codes is a super-set of the class of lossless instantaneous codes.

The interest in UD MASCs and SISCs is inspired by the fact that in traditional single-transmitter single-receiver single-source source coding, the set of achievable codeword lengths is the same for uniquely decodable and lossless instantaneous codes [38, 39]. Hence in searching for optimal code design algorithms, people focus on lossless instantaneous code design. For MASCs and SISCs, parallel results are not available yet.

In this section, I investigate various properties of UD MASCs and SISCs. In particular, I give necessary and sufficient conditions for UD MASCs and SISCs. I also give necessary conditions on the set of achievable codeword lengths for UD codes.

3.6.1 Necessary and Sufficient Conditions on Uniquely Decodable Multiple Access Source Codes

First, I would like to see if existing necessary and sufficient conditions for lossless instantaneous SISCs can be generalized to UD SISCs.

Define $\Gamma_y = \{\gamma_X(x) : p(x,y) > 0\}$ to be the codeword set for $\mathcal{A}_y = \{x \in \mathcal{X} : p(x,y) > 0\}$. Lemma 1 in Section 3.2 shows that code γ_X is a lossless instantaneous SISC for X given Y if and only if Γ_y is prefix-free for each $y \in \mathcal{Y}$. To generalize this result to UD SISCs, I conjecture that γ_X is a UD SISC for X given Y if and only if Γ_y is UD without side information for each $y \in \mathcal{Y}$.

The following lemmas show that this conjecture is only partially correct.

Lemma 6 *γ_X is a UD SISC for X given Y only if Γ_y is UD without side information for each $y \in \mathcal{Y}$.*

Proof. I would like to show that if there exists a $y^* \in \mathcal{Y}$, such that Γ_{y^*} is not UD, then there always exists a Y sequence $y_1, y_2, \ldots, y_K$ and two distinct X sequences $x_1, x_2, \ldots, x_K$ and $x'_1, x'_2, \ldots, x'_K$ with identical binary encoded strings

$$\gamma_X(x_1)\gamma_X(x_2)\ldots\gamma_X(x_K) = \gamma_X(x'_1)\gamma_X(x'_2)\ldots\gamma_X(x'_K),$$

and $p(x_i, y_i)p(x'_i, y_i) > 0$ for all $i \in [1, K]$.

Assume $\exists y^* \in \mathcal{Y}$, such that Γ_{y^*} is not UD. Then there exist two distinct X sequences $\mathbf{X}^K = (x_1, x_2, \ldots, x_K)$ and $\mathbf{X}^K = (x'_1, x'_2, \ldots, x'_K)$, such that $x_i, x'_i \in \{x \in \mathcal{X} : p(x, y^*) > 0\}$ for all $i \in [1, K]$ and $\gamma_X(x_1)\gamma_X(x_2)\ldots\gamma_X(x_K) = \gamma_X(x'_1)\gamma_X(x'_2)\ldots\gamma_X(x'_K)$. Then the

$Y \backslash X$	a	b	c
A	0.25	0.25	0
B	0.25	0	0.25
$\gamma_X(X)$	0	01	10

Table 3.1: Counterexample used for proving Lemma 7.

binary string $\gamma_X(x_1)\gamma_X(x_2)\ldots\gamma_X(x_K)$ with side information $\mathbf{Y}^K = (y^*, y^*, \ldots, y^*)$. cannot be uniquely decoded. □

Unfortunately, the necessary condition of Lemma 6 is not sufficient.

Lemma 7 *The condition that Γ_y be UD for each $y \in \mathcal{Y}$ is not sufficient to guarantee that γ_X is a UD SISC for X given Y.*

Proof. I use the following example. Let $p(x, y)$ and γ_X be given by Table 3.1. Note that both $\Gamma_A = \{\gamma_X : p(x, A) > 0\} = \{0, 01\}$ and $\Gamma_B = \{\gamma_X : p(x, B) > 0\} = \{0, 10\}$ are uniquely decodable. Consider Y sequence AB and encoded binary string 010 for X. String 010 can be parsed into two different X sequences ac and ba with $p(a, A)p(c, B) > 0$ and $p(b, A)p(a, B) > 0$. Hence, we cannot uniquely decode 010 given Y sequence AB. □

The example used in proving Lemma 7 can also be used to prove the following corollary.

Corollary 1 *If $\mathcal{Y} = \{A, B\}$, the condition that Γ_y be UD for each $y \in \mathcal{Y}$ and $\Gamma_A \cap \Gamma_B$ be prefix free is not sufficient to guarantee that γ_X is a UD SISC for X given Y.*

These results suggest that even for a very simple $p(x, y)$ and γ_X, we may need a more sophisticated procedure to test whether or not γ_X is UD. Theorems 11 and 12 generalize the test procedure of Sardinas and Patterson [40] to UD SISCs and UD MASCs.

If string $\mathbf{s}$ is the concatenation of two strings, i.e., $\mathbf{s} = \mathbf{s}_1\mathbf{s}_2$, then the operators $'+'$ and $'-'$ on strings are defined as $\mathbf{s} = \mathbf{s}_1 + \mathbf{s}_2$ and $\mathbf{s}_2 = \mathbf{s} - \mathbf{s}_1$. Thus '+' designates concatenation while '-' is a suffix operator.

Theorem 11 *SISC γ_X is UD given Y if and only if it passes TEST 1.*

TEST 1

1. Let $\mathcal{S}_0 = \{\gamma_X(x) : x \in \mathcal{X}\}$.

2. Let $\mathcal{S}_1 = \{\mathbf{c} - \mathbf{c}' : \mathbf{c}, \mathbf{c}' \in \Gamma_y \text{ for some } y \in \mathcal{Y} \text{ and } \mathbf{c}' \prec \mathbf{c}\}$ and $i = 1$.

3. For each pair $(\mathbf{c}, \mathbf{s})$ such that $\mathbf{c} \in \mathcal{S}_o$, $\mathbf{s} \in \mathcal{S}_i$, and $\mathbf{s} \prec \mathbf{c}$,

   (a) if $\mathbf{c}, \mathbf{c} - \mathbf{s} \in \Gamma_y$ for some $y \in \mathcal{Y}$, then γ_X fails the test and the procedure stops;

   (b) let $\mathcal{S}_{i+1}(\mathbf{c}, \mathbf{s}) = \{\mathbf{c}' - (\mathbf{c} - \mathbf{s}) : \mathbf{c}, \mathbf{c}' \in \Gamma_y \text{ for some } y \in \mathcal{Y} \text{ and } \mathbf{c} - \mathbf{s} \prec \mathbf{c}'\}$, (see Figure 3.8(a));

   (c) let $\dot{\mathcal{S}}_{i+1}(\mathbf{c}, \mathbf{s}) = \{(\mathbf{c} - \mathbf{s}) - \mathbf{c}' : \mathbf{c}, \mathbf{c}' \in \Gamma_y \text{ for some } y \in \mathcal{Y} \text{ and } \mathbf{c}' \prec \mathbf{c} - \mathbf{s}\}$, (see Figure 3.8(b)).

4. For each pair $(\mathbf{c}, \mathbf{s})$ such that $\mathbf{c} \in \mathcal{S}_o$, $\mathbf{s} \in \mathcal{S}_i$, and $\mathbf{c} \prec \mathbf{s}$, let $\ddot{\mathcal{S}}_{i+1}(\mathbf{c}, \mathbf{s}) = \{(\mathbf{s} - \mathbf{c}) + \mathbf{c}' : \mathbf{c}, \mathbf{c}' \in \Gamma_y \text{ for some } y \in \mathcal{Y}\}$ (see Figure 3.8(c)).

5. Let

$$\mathcal{S}_{i+1} = \cup_{\mathbf{c} \in \mathcal{S}_0} \left[\cup_{\mathbf{s} \in \mathcal{S}_i : \mathbf{s} \prec \mathbf{c}} \left[\mathcal{S}_{i+1}(\mathbf{c}, \mathbf{s}) \cup \dot{\mathcal{S}}_{i+1}(\mathbf{c}, \mathbf{s}) \right] \cup_{\mathbf{s} \in \mathcal{S}_i : \mathbf{c} \prec \mathbf{s}} \ddot{\mathcal{S}}_{i+1}(\mathbf{c}, \mathbf{s}) \right].$$

6. If every string in $\mathcal{S}_{i+1}$ has appeared in some $\mathcal{S}_j$, $j \leq i$ or $\mathcal{S}_{i+1}$ is empty, then γ_X passes the test, and the procedure stops. Otherwise, increment i and go to step 4.

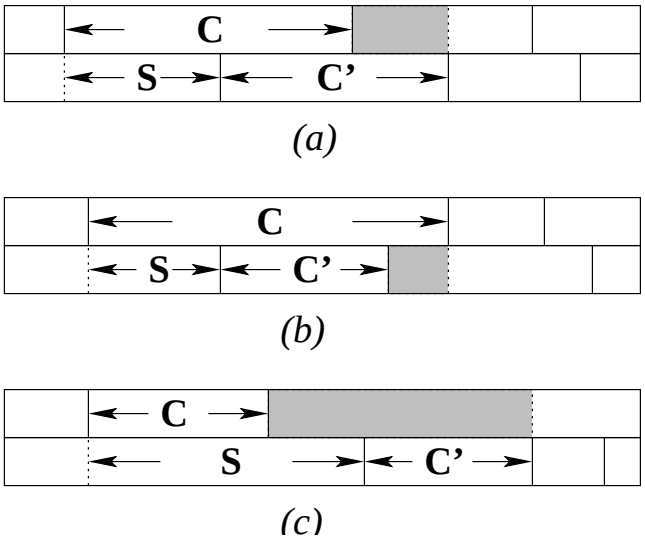


Figure 3.8: Three prefix/suffix relations between strings $\mathbf{c} \in \mathcal{S}_o$ and $\mathbf{s} \in \mathcal{S}_i$.

Table 3.2 gives an example of how to use TEST 1 to determine if γ_X is a UD SISC. The test fails in the construction of $\mathcal{S}_3$ since $\mathbf{c} = 100 \in \mathcal{S}_0$, $\mathbf{s} = 1 \in \mathcal{S}_2$ and $\mathbf{c_s} = 00$ implies $(\mathbf{c}, \mathbf{c} - \mathbf{s}) \in \Gamma_3$. Constructing a string that can be decoded in two distinct ways involves tracing back any path in the test that ends in a violation. In this example, the binary description 0101100 can be parsed as 010,11,00 or 01,01,100 when $(Y_1, Y_2, Y_3) = (2, 2, 3)$.

TEST 1 can be generalized to give a necessary and sufficient condition for UD MASCs.

Theorem 12 *MASC $\{\gamma_X, \gamma_Y\}$ is UD if and only if it passes TEST 2.*

First, I define a *two-step prefix/suffix procedure.* Figure 3.8 illustrates this process as well. The input is a set of codewords $\mathcal{C}$ and a suffix $\mathbf{s}$; the output is a collection of suffixes $\mathcal{S}$. The procedure works as follows. Given a suffix $\mathbf{s}$ and set of codewords $\mathcal{C}$, find all $\mathbf{c}, \mathbf{c}' \in \mathcal{C}$, such that

1. $\mathbf{s} \prec \mathbf{c}$ (step 1), $\mathbf{c} - \mathbf{s} \prec \mathbf{c}'$ (step 2). Put the suffix $\mathbf{s}' = \mathbf{c}' - (\mathbf{c} - \mathbf{s})$ (shaded string in Figure 3.8(a)) into $\mathcal{S}$.

Table 3.2: An example where γ_X is not UD SISC as tested by TEST 1. In this table, $\mathcal{S}_i$ are classified according to y, hence $\mathcal{S}_i$ may contain repeated elements. This classification is convenient but not necessary.

Γ_y	$\mathcal{S}_0$	$\mathcal{S}_1$	$\mathcal{S}_2$	$\mathcal{S}_3$
Γ_1	00		0	0
	01		1	1
	10			
	11			
Γ_2	00	0	0	0
	01		1	1
	11		10	10
	010			
Γ_3	00	0	0	0
	10			1
	11			00
	100			

2. $\mathbf{s} \prec \mathbf{c}$ (step 1), $\mathbf{c}' \prec \mathbf{c} - \mathbf{s}$ (step 2). Put the suffix $\mathbf{s}' = (\mathbf{c} - \mathbf{s}) - \mathbf{c}'$ (shaded string in Figure 3.8(b)) into $\mathcal{S}$.

3. $\mathbf{c} \prec \mathbf{s}$ (step 1), $\mathbf{c}' \in \mathcal{C}$ (step 2). Put $\mathbf{s}' = (\mathbf{s} - \mathbf{c}) + \mathbf{c}'$ (shaded string in Figure 3.8(c)) into $\mathcal{S}$.

TEST 2

1. Let $\mathcal{S}_{X0} = \{\gamma_X(x) : x \in \mathcal{X}\}$ and $\mathcal{S}_{Y0} = \{\gamma_Y(y) : y \in \mathcal{Y}\}$. Draw an undirected graph with vertex set $\mathcal{S}_{X0} \cup \mathcal{S}_{Y0}$ and edge set $\{(\gamma_X(x), \gamma_Y(y)) : p(x, y) > 0\}$.

2. $\mathcal{S}_{X1} \times \mathcal{S}_{Y1} = \{(\mathbf{c}_x - \mathbf{c}'_x) \times (\mathbf{c}_y - \mathbf{c}'_y) : \mathbf{c}_x, \mathbf{c}'_x \in \mathcal{S}_{X0}$ and $\mathbf{c}_y, \mathbf{c}'_y \in \mathcal{S}_{Y0}, \mathbf{c}'_x \prec \mathbf{c}_x$ and $\mathbf{c}'_y \prec \mathbf{c}_y, p(\mathbf{c}_x, \mathbf{c}_y)p(\mathbf{c}'_x, \mathbf{c}'_y) + p(\mathbf{c}_x, \mathbf{c}'_y)p(\mathbf{c}'_x, \mathbf{c}_y) > 0.\}$ [6] Connect $\mathbf{s}_x \in \mathcal{S}_{X1}$ with $\mathbf{s}_y \in \mathcal{S}_{Y1}$. Set $i = 1$.

3. If there exist codewords $\mathbf{c}_x, \mathbf{c}'_x \in \mathcal{S}_{X0}$, $\mathbf{c}_y, \mathbf{c}'_y \in \mathcal{S}_{Y0}$ and connected suffixes $\mathbf{s}_x \in \mathcal{S}_{Xi}, \mathbf{s}_y \in \mathcal{S}_{Yi}$ such that $p(\mathbf{c}_x, \mathbf{c}_y)p(\mathbf{c}'_x, \mathbf{c}'_y) + p(\mathbf{c}_x, \mathbf{c}'_y)p(\mathbf{c}'_x, \mathbf{c}_y) > 0$ and $\mathbf{s}_x = \mathbf{c}_x - \mathbf{c}'_x$, $\mathbf{s}_y = \mathbf{c}_y - \mathbf{c}'_y$, then (γ_X, γ_Y) fails the test and procedure stops.

4. Do the two-step prefix/suffix procedure for input codeword set $\mathcal{S}_{X0}$ and each suffix $\mathbf{s} \in \mathcal{S}_{Xi}$. Let $S_{X(i+1)}$ be the union of the output of that procedure. Similarly, running the procedure on $\mathcal{S}_{Y0}$ for each $\mathbf{s} \in \mathcal{S}_{Yi}$ gives $\mathcal{S}_{Y(i+1)}$.

5. Connect suffix $\mathbf{s}'_x \in \mathcal{S}_{X(i+1)}$ with suffix $\mathbf{s}'_y \in \mathcal{S}_{Y(i+1)}$ if and only if there exist codewords $\mathbf{c}_x, \mathbf{c}'_x \in \mathcal{S}_{X0}$, $\mathbf{c}_y, \mathbf{c}'_y \in \mathcal{S}_{Y0}$ and connected suffixes $\mathbf{s}_x \in \mathcal{S}_{Xi}, \mathbf{s}_y \in \mathcal{S}_{Yi}$ such that $p(\mathbf{c}_x, \mathbf{c}_y)p(\mathbf{c}'_x, \mathbf{c}'_y) + p(\mathbf{c}_x, \mathbf{c}'_y)p(\mathbf{c}'_x, \mathbf{c}_y) > 0$, codewords $\mathbf{c}_x, \mathbf{c}'_x$ and suffix $\mathbf{s}_x$ result in suffix

[6] Here $p(\mathbf{c}_x, \mathbf{c}_y) > 0$ if and only if $\exists (x, y) \in \mathcal{X} \times \mathcal{Y}$, such that $\gamma_X(x) = \mathbf{c}_x, \gamma_Y(y) = \mathbf{c}_y$ and $p(x, y) > 0$.

$\mathbf{s}'_x$, codewords $\mathbf{c}_y, \mathbf{c}'_y$ and suffix $\mathbf{s}_y$ result in suffix $\mathbf{s}'_y$, through two-step prefix/suffix processes respectively.

6. Remove all un-connected suffixes from $\mathcal{S}_{X(i+1)}$ and $\mathcal{S}_{Y(i+1)}$.

7. If $\mathcal{S}_{X(i+1)}$, $\mathcal{S}_{Y(i+1)}$ and their connecting pattern have appeared previously or at least one of $\mathcal{S}_{X(i+1)}$ or $\mathcal{S}_{Y(i+1)}$ is empty, then (γ_X, γ_Y) passes the test, and the procedure stops. Otherwise, increment i and repeat step 3.

3.6.2 Necessary Conditions on the Codeword Lengths for Uniquely Decodable Side Information Source Code

Section 3.6.1 describes testing procedures for determining if a particular code is UD. In this section, I generalize the Kraft Inequality [38, 39] to give necessary conditions on the codeword lengths for UD SISCs. Necessary and sufficient conditions are not yet known.

A set of necessary conditions on the codeword lengths for UD SISC can be easily obtained from Lemma 6 by applying the Kraft Inequality from traditional UD codes.

Corollary 2 *For every UD SISC on X given Y,*

$$\sum_{c \in \Gamma_y} 2^{-|c|} \leq 1 \quad \textit{for each} \quad y \in \mathcal{Y}.$$

I would like to seek more stringent necessary conditions. I proceed in order of increasing alphabet size of $\mathcal{Y}$ and finally generalize our results to arbitrary $|\mathcal{Y}|$. Lemmas 8 and 9 generalize two results from lossless instantaneous SISCs given in [23] to UD SISCs. Without loss of generality, let $\mathcal{Y} = \{1, \ldots, N_Y\}$, where $N_Y = |\mathcal{Y}|$.

Lemma 8 *If* $|\mathcal{Y}| = 2$, *then*

$$\sum_{c \in \Gamma_y} 2^{-|c|} \leq 1 \quad \textit{for each} \quad y \in \mathcal{Y}$$

is necessary and sufficient for the existence of UD SISC.

Proof. By Corollary 2 and [23, Theorem 1].

Note that Theorem 1 of [23] and Lemma 8 imply that the set of achievable codeword lengths is the same for UD SISC and for lossless instantaneous SISC when $|\mathcal{Y}| = 2$.

Lemma 9 *For every UD SISC on* X *given* Y *with* $|\mathcal{Y}| = 3$, $(\Gamma_1 \cap \Gamma_2) \cup (\Gamma_1 \cap \Gamma_3) \cup (\Gamma_2 \cap \Gamma_3)$ *is UD without side information.*

Proof. Assume $\Gamma^* = (\Gamma_1 \cap \Gamma_2) \cup (\Gamma_1 \cap \Gamma_3) \cup (\Gamma_2 \cap \Gamma_3)$ is not UD without side information. Then there exists an encoded binary string γ^* such that $\gamma^* = \gamma_X(x_1)\gamma_X(x_2)\ldots\gamma_X(x_K) = \gamma_X(x'_1)\gamma_X(x'_2)\ldots\gamma_X(x'_K)$, where $x_i, x'_i \in (\mathcal{A}_1 \cap \mathcal{A}_2) \cup (\mathcal{A}_1 \cap \mathcal{A}_3) \cup (\mathcal{A}_2 \cap \mathcal{A}_3)$ for all $i \in [1, K]$ and $x_1 \neq x'_1$.

Since $x_i, x'_i \in (\mathcal{A}_1 \cap \mathcal{A}_2) \cup (\mathcal{A}_1 \cap \mathcal{A}_3) \cup (\mathcal{A}_2 \cap \mathcal{A}_3)$, there exists a $y_i^* \in \{1, 2, 3\}$ such that both $p(x_i, y_i^*) > 0$ and $p(x'_i, y_i^*) > 0$, $i = 1, \ldots, K$. Let $\mathbf{Y}^* = (y_1^*, y_1^*, \ldots, y_K^*)$. Then γ^* cannot be uniquely decoded given side information $\mathbf{Y}^*$.

□

Theorem 13 *For any UD SISC on* X *given* Y,

$$(\Gamma_a \cap \Gamma_b) \cup (\Gamma_a \cap \Gamma_c) \cup (\Gamma_b \cap \Gamma_c)$$

is UD without side information for every $\{a, b, c\} \subseteq \mathcal{Y}$.

Lemma 6 is a special case of Theorem 13 with $a = b = c$.

Notice that the structure of Theorem 13 is critical to the result. For example, when $|\mathcal{Y}| = 4$, it is not necessary for $(\Gamma_1 \cap \Gamma_2) \cup (\Gamma_3 \cap \Gamma_4)$ be UD. Table 6.3 in the Appendix gives an example where none of $(\Gamma_1 \cap \Gamma_2) \cup (\Gamma_3 \cap \Gamma_4)$ or $(\Gamma_1 \cap \Gamma_3) \cup (\Gamma_2 \cap \Gamma_4)$ or $(\Gamma_1 \cap \Gamma_4) \cup (\Gamma_2 \cap \Gamma_3)$ is UD, but $\gamma_X(X)$ is a UD SISC for X given Y.

Corollary 3 *For any UD SISC on X given Y,*

$$\sum_{c \in (\Gamma_a \cap \Gamma_b) \cup (\Gamma_a \cap \Gamma_c) \cup (\Gamma_b \cap \Gamma_c)} 2^{-|c|} \leq 1$$

for every $\{a, b, c\} \subseteq \mathcal{Y}$.

Finally, we note that the results of [23] for $|\mathcal{Y}| = 2$ and $|\mathcal{Y}| = 3$ extend similarly to larger alphabets $\mathcal{Y}$.

Theorem 14 *For any lossless instantaneous SISC on X given Y,*

$$(\Gamma_a \cap \Gamma_b) \cup (\Gamma_a \cap \Gamma_c) \cup (\Gamma_b \cap \Gamma_c)$$

is prefix free for every $\{a, b, c\} \subseteq \mathcal{Y}$.

From the necessary conditions of Corollaries 3 I derive a lower bound on the rates achievable for by UD and lossless instantaneous SISCs. The following results apply when $|\mathcal{Y}| \leq 3$. (The proof of Theorem 15 is straight forward from Theorem 16.) The approach generalizes for larger alphabets.

Theorem 15 *For $|\mathcal{Y}| = 2$, the optimal rate $R(X)$ for a one-dimensional lossless SISC on X given Y satisfies $R^\star(X) \leq R(X) < R^\star(X) + 1$, where*

$$R^\star(X) = H(X) - (P_{1\bar{2}} + P_{\bar{1}2}) h\left(\frac{P_{1\bar{2}}}{P_{1\bar{2}} + P_{\bar{1}2}}\right),$$

and $P_{1\bar{2}} = \sum_{x \in \mathcal{A}_1 \cap \mathcal{A}_2^c} p(x)$, $P_{\bar{1}2} = \sum_{x \in \mathcal{A}_1^c \cap \mathcal{A}_2} p(x)$, $h(p) = -p \log p - (1-p)\log(1-p)$.

By using block coding, we can achieve $R^\star(X)$ asymptotically as the blocklength approaches infinity. I also show that $R^\star(X)$ is always bounded from below by $H(X|Y)$.

Lemma 10 *If* $a+b+c+d=1$ *for some non-negative* a, b, c, d, *then*

$$(a+b)\log(a+b) + (c+d)\log(c+d) + (a+c)\log(a+c) + (b+d)\log(b+d) \quad (3.1)$$

$$\leq \quad a\log a + b\log b + c\log c + d\log d \quad (3.2)$$

Proof. By Jenson's inequality, we have

$$\begin{aligned} & (3.1)-(3.2) \\ = \quad & a\log\frac{(a+b)(a+c)}{a} + b\log\frac{(a+b)(b+d)}{b} + c\log\frac{(c+d)(a+c)}{c} + d\log\frac{(c+d)(b+d)}{d} \\ \leq \quad & \log((a+b)(a+c) + (a+b)(b+d) + (c+d)(a+c) + (c+d)(b+d)) \\ = \quad & 0 \end{aligned}$$

Lemma 11 *For* $|\mathcal{Y}| = 2$, $R^\star(X) \geq H(X|Y)$, *with equality if and only if*

$$\begin{aligned} & \frac{p(x,1)}{p(x,2)} \text{ is a constant for all } x \in \mathcal{A}_1 \cap \mathcal{A}_2 \text{ and} \\ & \frac{\sum_{x \in \mathcal{A}_1 \cap \mathcal{A}_2} p(x,1)}{\sum_{x \in \mathcal{A}_1 \cap \mathcal{A}_2^c} p(x,1)} = \frac{\sum_{x \in \mathcal{A}_1 \cap \mathcal{A}_2} p(x,2)}{\sum_{x \mathcal{A}_1^c \cap \mathcal{A}_2} p(x,2)}. \end{aligned}$$

Proof. Let $A_{1\bar{2}} = \mathcal{A}_1 \cap \mathcal{A}_2^c$, $A_{\bar{1}2} = \mathcal{A}_1^c \cap \mathcal{A}_2$, $A_{12} = \mathcal{A}_1 \cap \mathcal{A}_2$, $P_{1\bar{2}} = \sum_{x \in A_{1\bar{2}}} p(x,1)$, $P_{\bar{1}2} = \sum_{x \in A_{\bar{1}2}} p(x,2)$, $P_{121} = \sum_{x \in A_{12}} p(x,1)$, $P_{122} = \sum_{x \in A_{12}} p(x,2)$, $P_{12} = P_{121} + P_{122}$.

We have

$$R^\star = -\sum_{x \in A_{1\bar{2}}} p(x,1)\log p(x,1) - \sum_{x \in A_{\bar{1}2}} p(x,2)\log p(x,2)$$

$$
\begin{aligned}
& - \sum_{x\in A_{12}} p(x,1)\log(p(x,1)+p(x,2)) - \sum_{x\in A_{12}} p(x,2)\log(p(x,1)+p(x,2)) \\
& + P_{1\bar{2}}\log P_{1\bar{2}} + P_{\bar{1}2}\log P_{\bar{1}2} - (P_{1\bar{2}}+P_{\bar{1}2})\log(P_{1\bar{2}}+P_{\bar{1}2})
\end{aligned}
$$

$$
\begin{aligned}
H(X|Y) &= - \sum_{x\in A_{1\bar{2}}} p(x,1)\log p(x,1) - \sum_{x\in A_{\bar{1}2}} p(x,2)\log p(x,2) \\
& - \sum_{x\in A_{12}} p(x,1)\log p(x,1) - \sum_{x\in A_{12}} p(x,2)\log p(x,2) \\
& + (P_{1\bar{2}}+P_{121})\log(P_{1\bar{2}}+P_{121}) + (P_{\bar{1}2}+P_{122})\log(P_{\bar{1}2}+P_{122})
\end{aligned}
$$

$$
\begin{aligned}
H(X|Y)-R^\star &= \sum_{x\in A_{12}} p(x,1)\log\frac{p(x,1)+p(x,2)}{p(x,1)} + \sum_{x\in A_{12}} p(x,2)\log\frac{p(x,1)+p(x,2)}{p(x,2)} \\
& + (P_{1\bar{2}}+P_{121})\log(P_{1\bar{2}}+P_{121}) + (P_{\bar{1}2}+P_{122})\log(P_{\bar{1}2}+P_{122}) \\
& - P_{1\bar{2}}\log P_{1\bar{2}} - P_{\bar{1}2}\log P_{\bar{1}2} + (P_{1\bar{2}}+P_{\bar{1}2})\log(P_{1\bar{2}}+P_{\bar{1}2}) \\
&\le P_{121}\log\sum_{x\in A_{12}}\frac{p(x,1)+p(x,2)}{P_{121}} + P_{122}\log\sum_{x\in A_{12}}\frac{p(x,1)+p(x,2)}{P_{122}} \\
& + (P_{1\bar{2}}+P_{121})\log(P_{1\bar{2}}+P_{121}) + (P_{\bar{1}2}+P_{122})\log(P_{\bar{1}2}+P_{122}) \\
& - P_{1\bar{2}}\log P_{1\bar{2}} - P_{\bar{1}2}\log P_{\bar{1}2} + (P_{1\bar{2}}+P_{\bar{1}2})\log(P_{1\bar{2}}+P_{\bar{1}2}) \qquad (3.3) \\
&= P_{121}\log\frac{P_{12}}{P_{121}} + P_{122}\log\frac{P_{12}}{P_{122}} \\
& - P_{1\bar{2}}\log P_{1\bar{2}} - P_{\bar{1}2}\log P_{\bar{1}2} + (P_{1\bar{2}}+P_{\bar{1}2})\log(P_{1\bar{2}}+P_{\bar{1}2}) \\
& + (P_{1\bar{2}}+P_{121})\log(P_{1\bar{2}}+P_{121}) + (P_{\bar{1}2}+P_{122})\log(P_{\bar{1}2}+P_{122}) \\
&= (P_{121}+P_{122})\log(P_{121}+P_{122}) + (P_{1\bar{2}}+P_{\bar{1}2})\log(P_{1\bar{2}}+P_{\bar{1}2}) \\
& + (P_{1\bar{2}}+P_{121})\log(P_{1\bar{2}}+P_{121}) + (P_{\bar{1}2}+P_{122})\log(P_{\bar{1}2}+P_{122}) \\
& - P_{121}\log P_{121} - P_{122}\log P_{122} - P_{1\bar{2}}\log P_{1\bar{2}} - P_{\bar{1}2}\log P_{\bar{1}2} \\
&\le 0. \qquad (3.4)
\end{aligned}
$$

In the above, (3.3) comes from Jenson's inequality and equality holds if and only if

$$\frac{p(x,1)+p(x,2)}{p(x,1)} \text{ and } \frac{p(x,1)+p(x,2)}{p(x,2)} \text{ are constants for all } x\in A_{12};$$

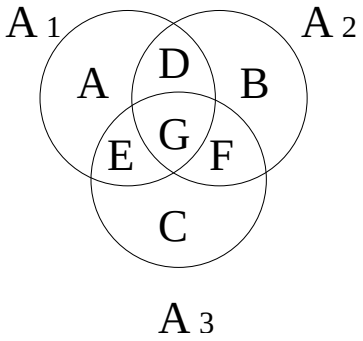


Figure 3.9: Definition of A, B, C, D, E, F, G.

and (3.4) comes from Lemma 10 with equality if and only if

$$\begin{cases} (P_{121} + P_{122})(P_{121} + P_{1\bar{2}}) = P_{121} \\ (P_{121} + P_{122})(P_{122} + P_{\bar{1}2}) = P_{122} \\ (P_{1\bar{2}} + P_{\bar{1}2})(P_{121} + P_{1\bar{2}}) = P_{1\bar{2}} \\ (P_{1\bar{2}} + P_{\bar{1}2})(P_{122} + P_{\bar{1}2}) = P_{\bar{1}2}. \end{cases}$$

Hence equality holds if and only if $p(x,1)/p(x,2)$ is constant for all $x \in A_{12}$ and $P_{121}/P_{1\bar{2}} = P_{122}/P_{\bar{1}2}$. □

The minimal value of $R^\star$ (which equals the minimal value of $H(X|Y)$) is $H(X) - 1$, achieved when $P_{12} = 0$ and $P_{1\bar{2}} = P_{\bar{1}2} = 1/2$.

When $|\mathcal{Y}| = 3$, to simplify our notation, we define (see Figure 3.9):

$$A = \mathcal{A}_1 \cap \mathcal{A}_2^c \cap \mathcal{A}_3^c,$$

$$B = \mathcal{A}_1^c \cap \mathcal{A}_2 \cap \mathcal{A}_3^c,$$

$$C = \mathcal{A}_1^c \cap \mathcal{A}_2^c \cap \mathcal{A}_3,$$

$$D = \mathcal{A}_1 \cap \mathcal{A}_2 \cap \mathcal{A}_3^c,$$

$$E = \mathcal{A}_1 \cap \mathcal{A}_2^c \cap \mathcal{A}_3,$$

$$F = \mathcal{A}_1^c \cap \mathcal{A}_2 \cap \mathcal{A}_3,$$

$$G = \mathcal{A}_1 \cap \mathcal{A}_2 \cap \mathcal{A}_3,$$

$$\text{and} \quad P_S = \sum_{x \in S} p_X(x).$$

Theorem 16 *For $|\mathcal{Y}| = 3$, the optimal rate $R(X)$ for a one-dimensional lossless SISC on X given Y satisfies*

$$R(X) \geq H(X) - (P_A + P_F)h(\frac{P_A}{P_A + P_F}) - (P_B + P_E)h(\frac{P_B}{P_B + P_E}) - (P_C + P_D)h(\frac{P_C}{P_C + P_D}).$$

Proof. From the necessary conditions of Corollary 3, we can obtain a lower bound on $R(X)$ using Lagrangian minimization to minimize

$$\begin{aligned} J &= \sum_x p(x)l(x) \\ &\quad + \sum_{x \in A \cup D \cup E \cup G} \lambda_1' 2^{-l(x)} + \sum_{x \in B \cup D \cup F \cup G} \lambda_2' 2^{-l(x)} + \sum_{x \in C \cup E \cup F \cup G} \lambda_3' 2^{-l(x)} + \sum_{x \in D \cup E \cup F \cup G} \lambda_4' 2^{-l(x)} \end{aligned}$$

We set the derivative $\mathrm{d}J/\mathrm{d}l(x) = 0$, and obtain

$$p(x) = \begin{cases} \lambda_1 2^{-l(x)} & x \in A \\ \lambda_2 2^{-l(x)} & x \in B \\ \lambda_3 2^{-l(x)} & x \in C \\ (\lambda_1 + \lambda_2 + \lambda_4) 2^{-l(x)} & x \in D \\ (\lambda_1 + \lambda_3 + \lambda_4) 2^{-l(x)} & x \in E \\ (\lambda_2 + \lambda_3 + \lambda_4) 2^{-l(x)} & x \in F \\ (\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4) 2^{-l(x)} & x \in G \end{cases}$$

where $\lambda_i = \lambda_i'/\ln 2$. From $\sum_x p(x) = 1$, and assuming the necessary conditions are met with equality, i.e., $\sum_{x \in A \cup D \cup E \cup G} \lambda_1' 2^{-l(x)} = 1$, $\sum_{x \in B \cup D \cup F \cup G} \lambda_2' 2^{-l(x)} = 1$, $\sum_{x \in C \cup E \cup F \cup G} \lambda_3' 2^{-l(x)} =$

1, $\sum_{x\in D\cup E\cup F\cup G}\lambda_4' 2^{-l(x)}=1$, we get $\sum_{i=1}^{4}\lambda_i=1$ and

$$\begin{cases} P_A/\lambda_1 = P_F/(\lambda_2+\lambda_3+\lambda_4) \\ P_B/\lambda_2 = P_E/(\lambda_1+\lambda_3+\lambda_4) \\ P_C/\lambda_3 = P_D/(\lambda_1+\lambda_2+\lambda_4). \end{cases}$$

Hence

$$\begin{cases} \lambda_1 = P_A/(P_A+P_F) \\ \lambda_2 = P_B/(P_B+P_E) \\ \lambda_3 = P_C/(P_C+P_D) \\ \lambda_4 = 1-\lambda_1-\lambda_2-\lambda_3, \end{cases}$$

which gives

$$l(x)=\begin{cases} -\log(\frac{P_A+P_F}{P_A}p(x)) & x\in A \\ -\log(\frac{P_B+P_E}{P_B}p(x)) & x\in B \\ -\log(\frac{P_C+P_D}{P_C}p(x)) & x\in C \\ -\log(\frac{P_C+P_D}{P_D}p(x)) & x\in D \\ -\log(\frac{P_B+P_E}{P_E}p(x)) & x\in E \\ -\log(\frac{P_A+P_F}{P_F}p(x)) & x\in F \\ -\log(p(x)) & x\in G. \end{cases}$$

Thus $R'=\sum_x p(x)l(x)=H(X)-(P_A+P_F)h(\frac{P_A}{P_A+P_F})-(P_B+P_E)h(\frac{P_B}{P_B+P_E})-(P_C+P_D)h(\frac{P_C}{P_C+P_D})$ is a lower bound on $R(X)$.[7] □

[7]When $|\mathcal{Y}|=2$, we have $P_C=0$. The necessary conditions of Corollary 3 are also sufficient for the existence of a UD SISC, which gives the results of Theorem 15.

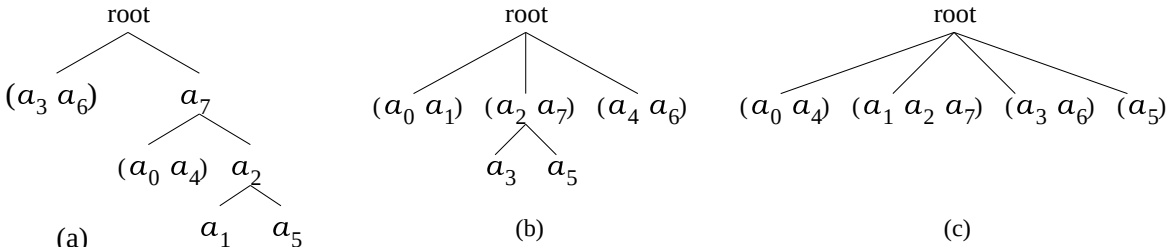


Figure 3.10: Partition trees for the p.m.f. from Table 6.1(a) using (a) optimal arithmetic coding, (b) optimal Huffman coding, (c) arithmetic or Huffman coding with the approach in [1].

3.7 Experimental Results

3.7.1 Optimal Design Algorithms

This section shows optimal coding rates for lossless SISCs, lossless MASCs, and near-lossless MASCs for the p.m.f.s of Tables 6.1 and 6.2 in the Appendix. I achieve these results by building the optimal partitions and matched codes for each scenario using the algorithms discussed in Sections 3.2, 3.3, and 3.4. Both Huffman and arithmetic coding rates are included.

Table 3.3 gives SISC results. As an example, for the p.m.f. in Table 6.1(a), the rate achievable in coding X using side information Y is approximately half that of an ordinary Huffman code and 90% that of [1]; the corresponding partition trees appear in Figure 3.10. The number of partitions tested to get the optimal rates ranges from $2|\mathcal{X}|^2$ to $|\mathcal{X}|^3$ for these p.m.f. examples.

Figure 3.11 shows general lossless and near-lossless MASC results compared with the corresponding bounds and the independent coding results. The optimal lossless MASC gives

Table	$H(X)$	$R'_{SI,A}(X)$	$R^{\star}_{SI,A}(X)$	$R_H(X)$	$R'_{SI,H}(X)$	$R^{\star}_{SI,H}(X)$
6.1(a)	2.91075	1.67976	1.53582	2.96	1.75	1.67
6.1(b)	2.51160	1.8201	1.79381	2.54	1.94	1.94
6.1(c)	2.91623	1.5071	1.46162	2.96	1.56	1.49
6.1(d)	2.91623	1.2784	1.15161	2.96	1.46	1.2
6.2(a)	3.85278	3.04098	2.94631	3.8764	3.07865	2.97472
6.2(b)	3.90508	3.27019	3.17971	3.93979	3.31152	3.21204

Table 3.3: Lossless SISC results for the p.m.f.s of Tables 6.1 and 6.2. In this table, $[H(X), R'_{SI,A}(X), R^{\star}_{SI,A}(X)]$ and $[R_H(X), R'_{SI,H}(X), R^{\star}_{SI,H}(X)]$ denote the optimal and Huffman results, respectively, for [traditional, SISC [1], optimal SISC] coding on X when Y is given as side information to the decoder.

significant performance improvement over independent coding of X and Y. For the example of Table 6.1(a), near-lossless coding with error probability 0.01 gives big improvements over lossless coding. The number of partitions tested in tracing out the given rate regions are bounded above by $2|\mathcal{X}|^4$, $|\mathcal{X}|^4$, $2|\mathcal{X}|^6$, $2|\mathcal{X}|^6$, $2|\mathcal{X}|^4$, and $|\mathcal{X}|^3/2$, respectively, for these six p.m.f.s.

Figure 3.12 shows the effect of the coding dimension on the achievable rate for a fixed error probability. I use the following p.m.f. Let $\mathcal{X} = \mathcal{Y} = \{0, 1\}$ and $p_Y(1) = 0.5$, $p_{X|Y}(0|0) = \alpha$, and $p_{X|Y}(1|1) = \beta$ for some $0 \leq \alpha, \beta < 0.5$. Figure 3.12 shows the near lossless MASC performance for $P_e \leq 5 \times 10^{-5}$ at parameter values $\alpha = \beta = 0.0002$ and $\alpha = 0.002$, $\beta = 0.0002$. As the coding dimension increases, the achievable rate region improves. The number of partitions tested in tracing out the rate regions is bounded above by 2^8 for dimension-3, $\epsilon = 5e - 5$ near-lossless MASCs.

Finally in Table 3.4, I compare the running time of (A) the optimal SISC design of Section 3.2 with (B) a later optimal design used in [32]. All experiments are run under identical conditions. Results are normalized to the running time of (A). While no general results are available, the comparison of (A) and (B) demonstrates the existence of examples where the more structured search presented in Section 3.2 reduces complexity relative to [32] very significantly.

3.7.2 Low Complexity Design Algorithms

I test both lossless and near-lossless SISC performance using the dynamic programming algorithm from Section 3.5.4.

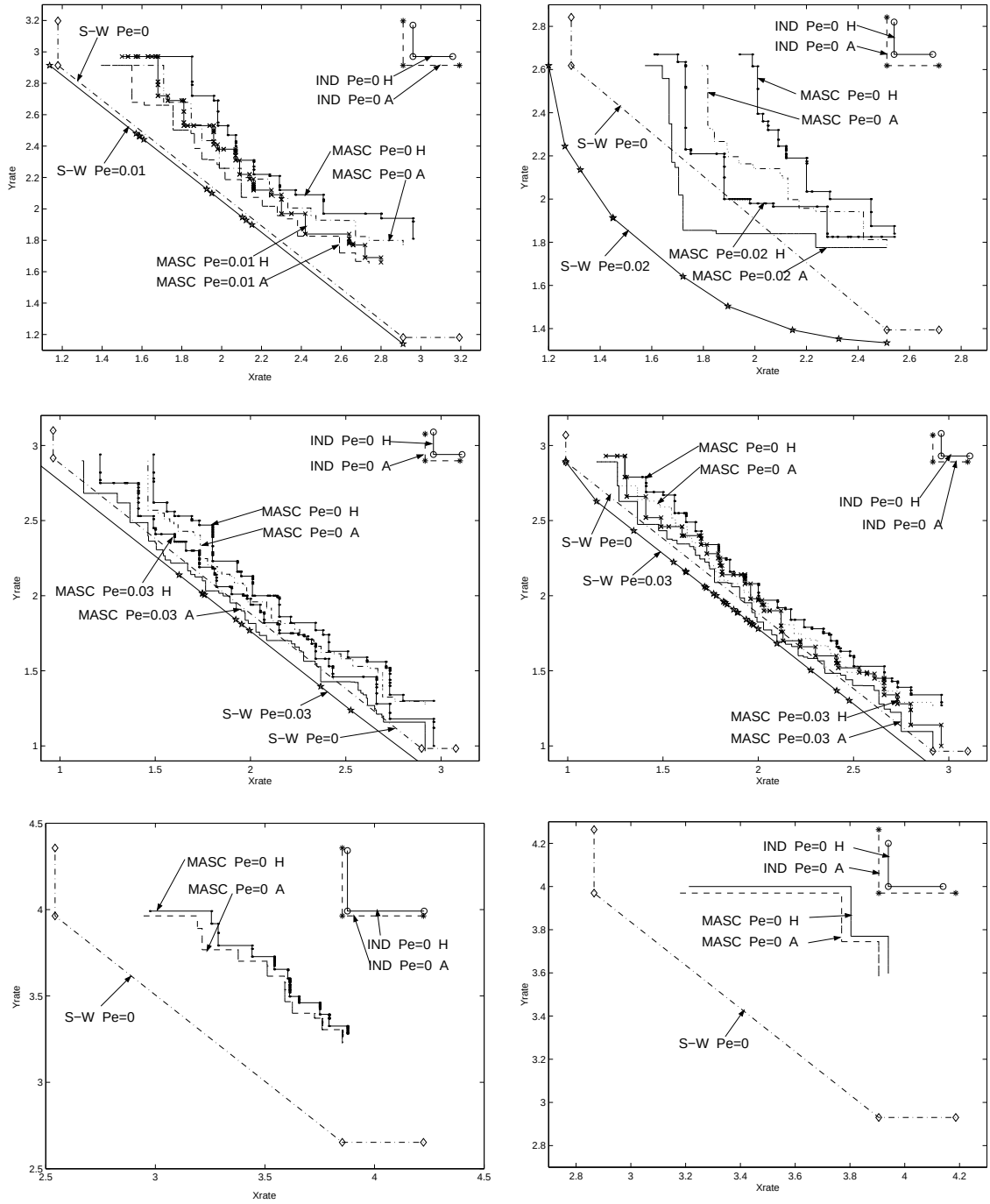
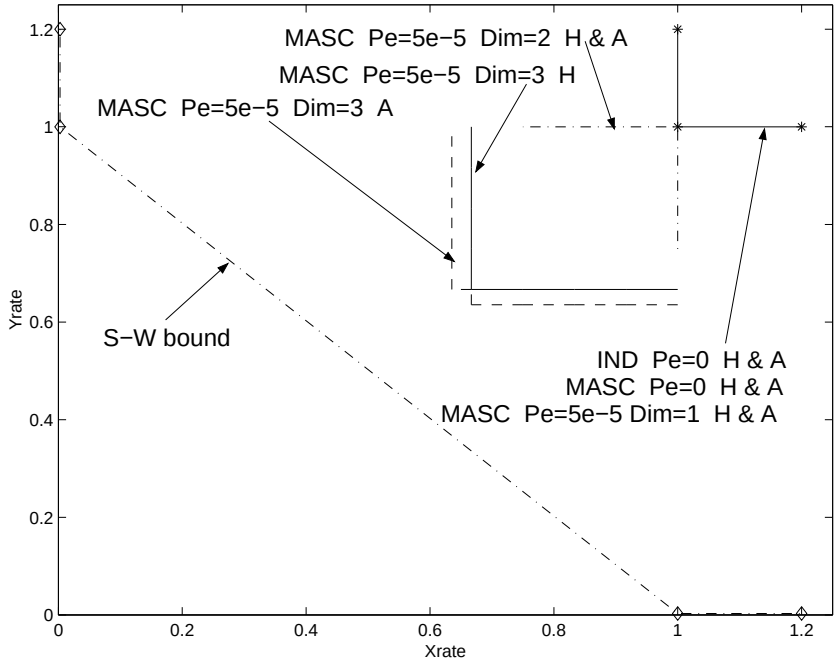
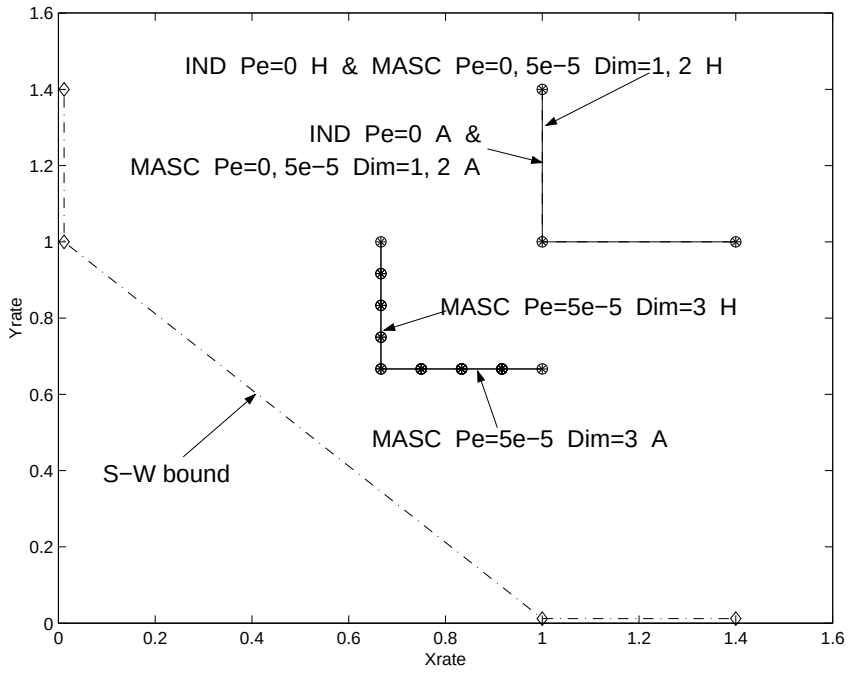


Figure 3.11: Dimension-1 lossless and near-lossless MASC results for Tables 6.1(a) (top left), 6.1(b) (top right), 6.1(c) (middle left), 6.1(d) (middle right), 6.2(a) (bottom left), and 6.2(b) (bottom right). (H: Huffman code; A: arithmetic code; $S-W, P_e = 0$: Slepian-Wolf bound; $S-W, P_e = \epsilon$: bound for near-lossless MASC)



(a)



(b)

Figure 3.12: Performance of near lossless MASC as a function of coding dimension (Dim), for (a) $\alpha = \beta = 0.0002$ and (b) $\alpha = 0.002$, $\beta = 0.0002$. (H: Huffman code; A: arithmetic code)

Table	6.1(a)	6.1(b)	6.1(c)	6.1(d)	6.2(a)	6.2(b)
SISC on X (A)	1	1	1	1	1	1
SISC on X (B)	13.056	11.40	20.56	3.681	1.26×10^5	1.36×10^6
SISC on Y (A)	1	1	1	1	1	1
SISC on Y (B)	7.672	15.20	3.664	3.661	8050.32	89145.32

Table 3.4: Comparing the running time of optimal design approaches.

Lossless SISC

For lossless SISCs, I use the following sources. Let $G_{N,q} = (\mathcal{X}, E_{\mathcal{X}})$ be a random graph with N vertices; each pair of vertices is connected with probability q (independent of all other pairs). I choose a distribution $P[\cdot]$ on the underlying size-N alphabet by drawing $u_1, \ldots, u_N$ uniformly on $(0, 1)$ and then normalizing.[8]

I test the fast algorithms on $G_{N,q}$ and P using $N \in \{8, 16, 32, 64, 128, 256\}$ and $q \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. I show the performance of our algorithm as a function of the number C of orders tested. Since the algorithm involves random order choices, I run each experiment K times. I measure the performance of the algorithm both by the fraction of trials in which the algorithm achieves the target rate (the globally optimal rate when N is small or the best rate from N^4 randomly chosen orders when N is large) and by the average (over trials) of the code's rate at the end of a trial.

[8]By [24], if pmfs p and p' satisfy (1) $p(x, y) = 0$ if and only if $p'(x, y) = 0$ and (2) $\sum_y p(x, y) = \sum_y p'(x, y)$ for all x, then the optimal lossless SISCs and expected performances for the two pmfs are identical. Therefore, I specify only $P[x]$ for all x and not $p(x, y)$ for all (x, y), and the performance on a single graph $G_{N,q}$ represents the performance on a large collection of sources.

In Figure 3.13, I present the experimental results as a function of C for $N \in \{16, 64\}$ and $q \in \{0.3, 0.7\}$ when a set of generally good parameters is used for each of the algorithms. Table 3.5 compares how close the fast algorithm's results come to the optimum. Table 3.6 gives the best rates the fast algorithms obtained for $N = 256$ and $q \in \{0.3, 0.5, 0.7\}$. Table 3.7 summarizes the parameters used in Figure 3.13 and Table 3.6.

From these figures and tables, I have the following observations:

1. SA is the most successful at avoiding local minima, but its rate of convergence is slow. When C is N^3, the average rate is close to the target rate, but the probability of hitting the target rate is still very low for $N \geq 16$.

2. The rate of DN decreases much faster than that of SA when the complexity is low. If the DN algorithm is run only once, it usually gets stuck with a local minimum. By running DN multiple times, we avoid this problem in all of our experiments and achieve a much better performance than SA. At $C = N^3$, multiple-run DN achieves performances very close to the optimum. For example, for $N = 16$, at $C = N^3$, the probability of hitting the optimal solution is at least 0.97 and the average rate differs from the optimal rate by at most 0.02%. Even at $C = N^2$ and $C = 2N^2$, the average rates differ from the optimal rate by at most 2% and 1%, respectively.

3. SA+DN and DN-merged-in-SA perform slightly better than SA but worse than DN for $N = 16$ and $q = 0.3, 0.5$; they perform better than DN for $N = 16$ and $q = 0.7$. For $N = 64$ and $C \leq 2N^2$, DN-merged-in-SA achieves performance very close to that of DN and both are much better than SA + DN's; but as the complexity further increases, DN-merged-in-SA outperforms the others, especially in terms of the probability of

hitting the target rate for $q = 0.5, 0.7$.

Near-lossless SISC

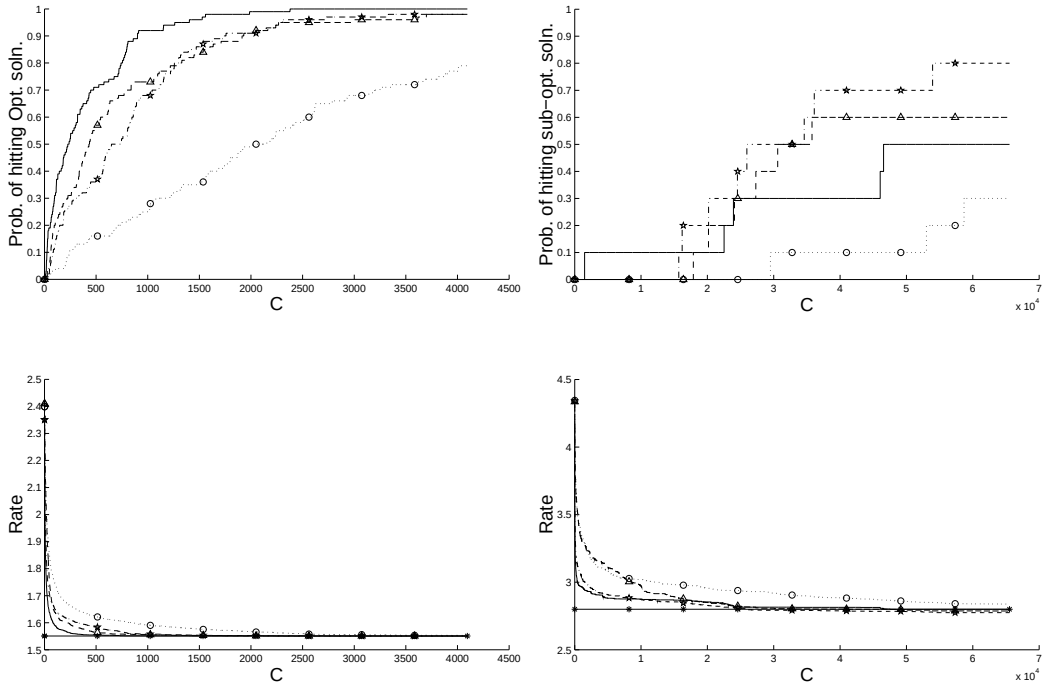
For near-lossless coding, we need to specify the exact $p(x, y)$ in order to calculate error probability P_e. Hence, I generate the random source as follows: assume a fixed percentage z of zero entries in $p(x, y)$; assign the designated number of zeros to randomly chosen entries; choose the value for the remaining entries at random; and finally normalize.

I use the same sets of parameters as in Table 3.7 for the four algorithms. For each fixed constant λ, near-lossless coding achieves similar performance to lossless coding in terms of the probability of hitting the target solution and the average value of the optimizing criteria $J_\lambda = R + \lambda P_e$ as a function of the number of orders searched.

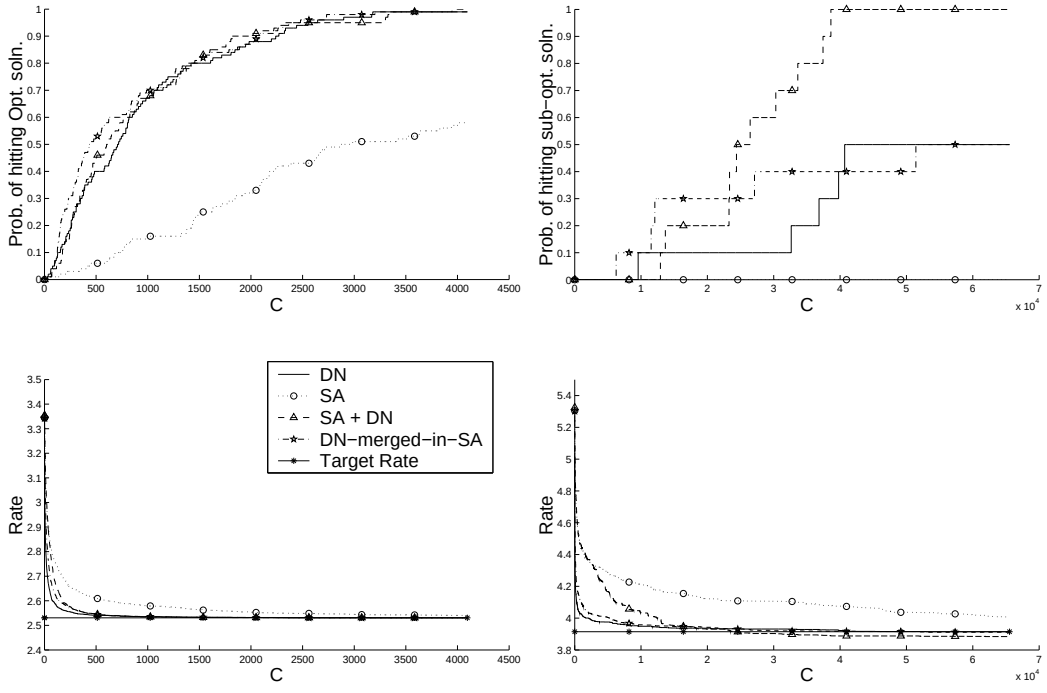
By varying the value of λ, I can trace out the curve between error probability P_e and rate R. Figure 3.14 shows the near-lossless coding results for three random sources. Two sets of curves corresponding to high design complexity (searching $C = N^3$ orders) and low design complexity (searching $C = N$ orders) are plotted for each source. From these plots, we can see that by allowing an error probability 0.1%, the rate can drop up to 3%. At high design complexity, the four algorithms achieve performance very close to each other; at low design complexity, the DN algorithm outperforms all the other algorithms significantly.

3.7.3 Universal Codes Design Algorithms

The low complexity sub-optimal SISC design algorithm can be applied in a variety of network coding scenarios to achieve low complexity network coding algorithms. One example is to apply low complexity near-lossless SISC design in network vector quantization [7].



(a) $N = 16$, $q = 0.3$ (c) $N = 64$, $q = 0.3$



(b) $N = 16$, $q = 0.7$ (d) $N = 64$, $q = 0.7$

Figure 3.13: Performance of four fast lossless SISC design algorithms.

Table 3.5: Closeness to the optimal solution. $Ratio = R_{fast}/R_{opt}$, R_{fast} = average rate of the fast algorithm over K trials, $Prob.$ = probability of hitting the target rate.

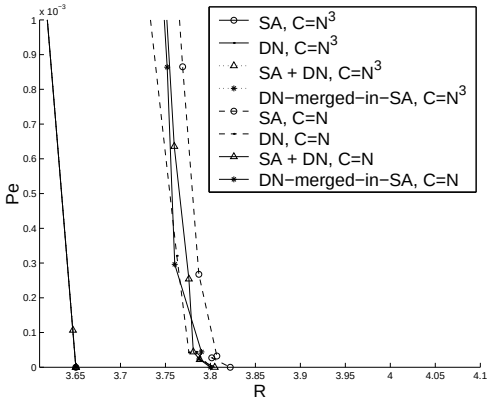
$N = 16$		$q = 0.3$		$q = 0.5$		$q = 0.7$	
		$Prob.$	$Ratio$	$Prob.$	$Ratio$	$Prob.$	$Ratio$
SA	$C = N^2$	0.11	1.0701	0.02	1.0874	0.03	1.0488
	$C = 2N^2$	0.16	1.0455	0.03	1.0610	0.06	1.0311
	$C = N^3$	0.79	1.0013	0.30	1.0147	0.58	1.0038
DN	$C = N^2$	0.54	1.0063	0.20	1.0222	0.19	1.0094
	$C = 2N^2$	0.71	1.0022	0.36	1.0121	0.40	1.0041
	$C = N^3$	1	1	0.97	1.0002	0.99	1.000
SA + DN	$C = N^2$	0.31	1.0261	0.12	1.0433	0.19	1.0173
	$C = 2N^2$	0.57	1.0084	0.21	1.0248	0.46	1.0055
	$C = N^3$	0.98	1	0.83	1.0015	1	1
DN-merged-in-SA	$C = N^2$	0.27	1.0394	0.06	1.0460	0.31	1.0154
	$C = 2N^2$	0.37	1.0213	0.11	1.0351	0.53	1.0050
	$C = N^3$	0.98	1	0.67	1.0050	0.99	1

Table 3.6: Achievable rates for $N = 256$, complexity limit $C = N^2 = 65536$.

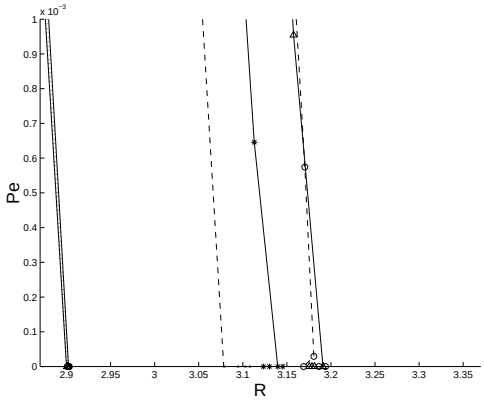
$N = 256$	q	SA	DN	SA + DN	DN-merged-in-SA
Huffman Rate	0.3	5.00656	4.43684	4.38234	4.30231
$= 7.75968$	0.5	5.57978	5.09944	5.06005	4.96012
	0.7	6.16025	5.61802	5.59836	5.53641

Table 3.7: Good parameters

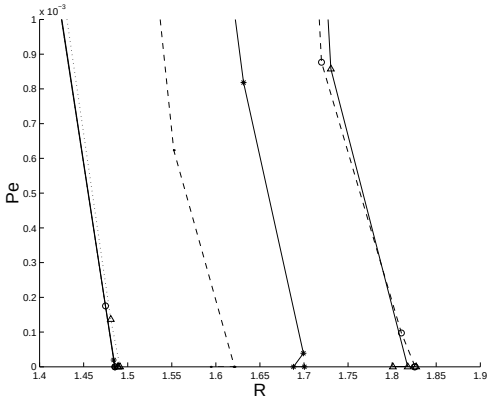
	SA	DN	SA + DN	DN-merged-in-SA
	$L_{in} = 4$	$L_{DN} = 16$	$L_{SA} = 4$	$L_{in} = 4$
$N = 16$ $(K = 100)$	$L_{out} = 16$	$p_a = 0$	$L_{DN} = 16$	$L_{out} = 16$
$C = 4096$	$T_o = 0.01$		$L_{out} = 4$	$P_s = 0.7$
			$T_o = 0.01$	$T_o = 0.01$
$N = 64$ $(K = 10)$	$L_{in} = 256$	$L_{DN} = 1024$	$L_{SA} = 256$	$L_{in} = 256$
$N = 256$ $(K = 1)$	$L_{out} = 256$	$p_a = 0$	$L_{DN} = 1024$	$L_{out} = 256$
$C = 65536$	$T_o = 0.01$		$L_{out} = 256$	$P_s = 0.7$
			$T_o = 0.01$	$T_o = 0.01$



(a) $N = 16$, $z = 50\%$



(b) $N = 16$, $z = 75\%$



(c) $N = 16$, $z = 90\%$

Figure 3.14: Performance of four fast near-lossless SISC design algorithms.

Reference [7] shows the concrete benefit of using variable-rate near-lossless SISC over (1) using variable-rate lossless SISC, (2) using fixed-rate lossless SISC and (3) using no side information in network vector quantization. Universal linked side information source codes (ULSISCs) [10] presents another application for SISC design. In this section, I give a brief introduction of ULSISC and present experimental results.

In Sections 3.2, 3.3, 3.4, and 3.5, I design optimal or sub-optimal SISCs and MASCs for a given joint source distribution $p(x, y)$. When $p(x, y)$ is unknown at design time, we require more sophisticated techniques to achieve good performance. A *universal linked side information source code* (ULSISC) [44, 45] is a modified SISC that achieves asymptotically optimal performance for any $p(x, y)$ on fixed alphabet $\mathcal{X} \times \mathcal{Y}$. The modification involves allowing an asymptotically negligible amount of communication from the decoder back to the encoder. This modification is critical in order to achieve universality in the SISC framework [44, 45].

The proof of the existence of ULSISCs given in [44, 45] is constructive. The encoder describes some fraction of the incoming data sequence to the decoder using a simple source code. The decoder then estimates $p(x, y)$ and describes its estimate to the encoder. Finally, the encoder describes X^n using an SISC matched to the distribution estimate. A careful balance between the fraction of the data sequence used in the distribution estimate and the resulting estimation accuracy allows the ULSISC to achieve asymptotically optimal performance. Details on ULSISC design appear in [10]. Without going into details of how to achieve a good balance, I here present only the experimental results using the set of parameters described in [10].

Due to the high complexity of optimal SISC design, I use the low complexity DN coding technique of Section 3.5 to design the sub-optimal SISC matched to the estimated distribu-

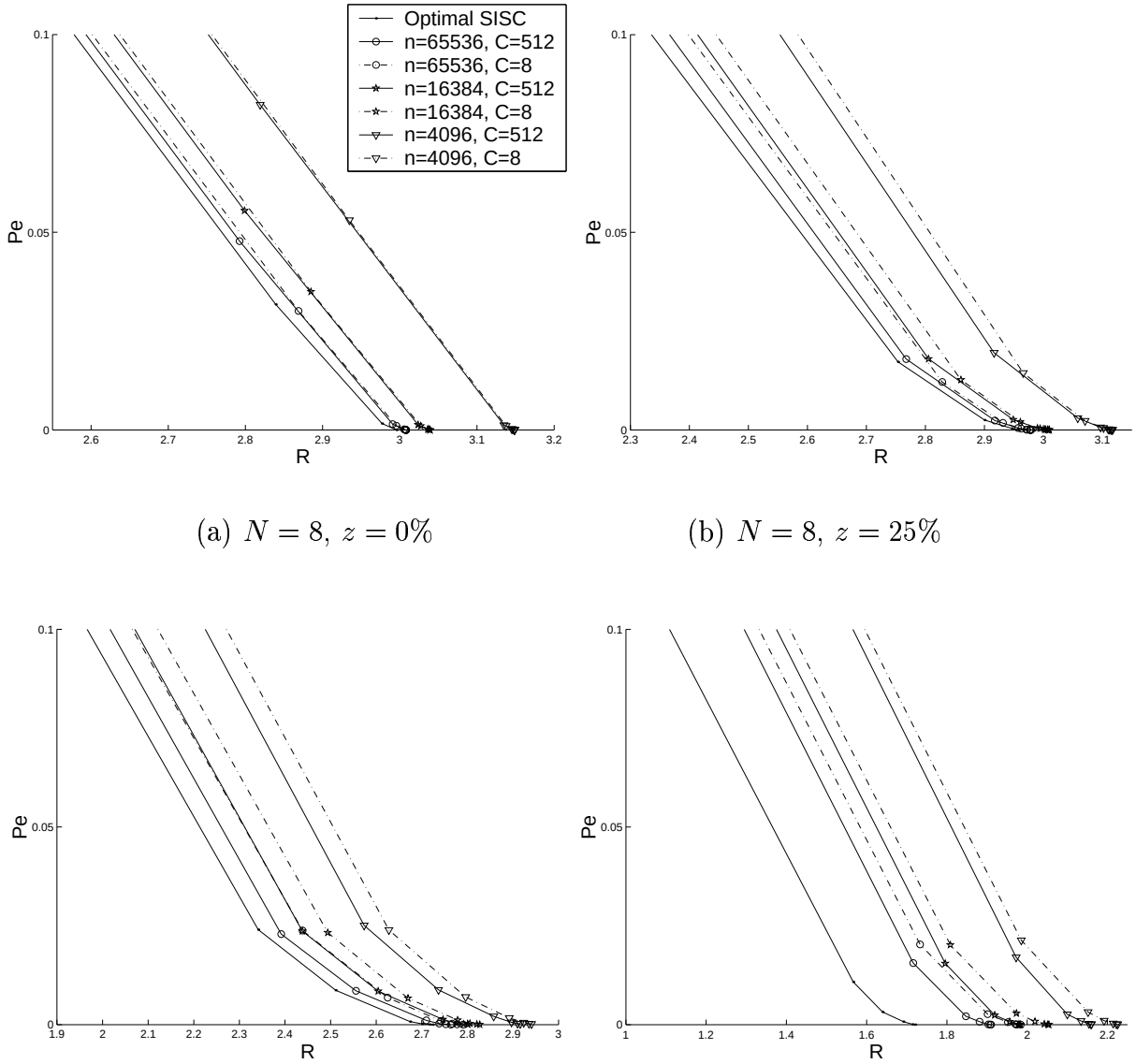
tion.

The experiments are performed on random sources generated in the same way as in Section 3.7.2, where I fix the percentage z of zero entries in $p(x, y)$, assign the designated number of zeros to randomly chosen entries, choose the value for the remaining entries at random, and normalize. By varying the value of λ I can trace out the curve between error probability P_e and rate R. I fix the set of parameters used in estimating $p(x, y)$ and vary the block length n and the SISC design complexity C. We compare the resulting ULSISC curves to each other and to the performance of the optimal SISC designed for the true $p(x, y)$.

Figure 3.15 shows the experimental results for four random sources. These results demonstrate that using low complexity SISC design in ULSISC when $p(x, y)$ is unknown can achieve performance very close to the optimal SISC performance when $p(x, y)$ is known. Increasing blocklength n has a greater impact on improving the performance than increasing the sub-optimal SISC design complexity.

3.8 Summary

This chapter demonstrates that the optimal lossless and near-lossless MASC design problems can be broken into two sub-problems: partition design and matched code design. The partition of an MASC describes the prefix and equivalence relationships for the code's binary descriptions. I give necessary and sufficient conditions on these partitions for instantaneous and lossless or near-lossless decoding and describe a variety of properties of the optimal partition that decrease the complexity associated with optimal partition design. I demonstrate the relationship between optimal matched codes and traditional (single-sender, single-receiver)



(a) $N = 8$, $z = 0\%$

(b) $N = 8$, $z = 25\%$

(c) $N = 8$, $z = 50\%$

(d) $N = 8$, $z = 75\%$

Figure 3.15: Performance of ULSISC, $N = |\mathcal{X}| = |\mathcal{Y}|$.

source codes and use this relationship to give optimal matched code design algorithms. When combined, these results characterize lossless and near-lossless SISCs and MASCs and yield a means of searching for the optimal codes of those types for an arbitrary source p.m.f. $p(x, y)$. Experimental results based on this algorithm are consistent with the theory.

Since optimal MASC code design is NP-hard, I provide a polynomial-time optimal solution for a constrained MASC design problem, then use this solution to solve the unconstrained MASC design problem and give a family of low complexity algorithms which approximate the optimal design for general p.m.f.s. Experimental results compare the achievable rates and error probability of different low complexity design algorithms with each other and with the optimal solution. These results demonstrate that the DN, SA+DN, and DN-merged-in-SA algorithms yield good performance for a wide range of source distributions; DN is better suited for low design complexity requirement, while SA+DN and DN-merged-in-SA are better suited for large alphabet size and high design complexity. The optimal and sub-optimal SISC design algorithm has been applied in universal linked side information source coding and in network vector quantizer design, demonstrating its applicability in various network source coding scenarios.

I also investigate the properties of uniquely decodable MASCs. I provide necessary and sufficient conditions for uniquely decodable MASCs which can be examined by a testing procedure. I further provide necessary conditions on the codeword lengths for uniquely decodable SISCs for arbitrary p.m.f.s.

Chapter 4

Entropy Constrained Dithered Quantization

4.1 Introduction

In general, to obtain a lossy code that achieves the asymptotically optimal performance requires high design complexity. Entropy constrained dithered quantization (ECDQ) [46, 47] is an exception; it is a simple procedure for source-independent lossy coding and can achieve performance very close to the theoretical optimal performance. I apply ECDQ to network coding environment in this chapter.

ECDQ is dithered uniform or lattice quantization followed by universal entropy coding. The ECDQ's universal entropy code (e.g., a Lempel-Ziv-based algorithm) captures the source statistics. We define the rate redundancy of an ECDQ with expected distortion D as the difference between the rate (measured as entropy) of the ECDQ and the rate-distortion function for the same source at the same distortion. In [47], Zamir and Feder give an

upper bound on the rate redundancy that holds for all K-dimensional vector sources and all distortion values and is constant for the squared-error distortion measure.

ECDQ

A uniform scalar quantizer $Q : \mathcal{R} \to \mathcal{R}$ is defined as $Q(x) = i\Delta$ for all $x \in (i\Delta - \Delta/2, i\Delta + \Delta/2]$. The dither random variable Z is uniformly distributed in $(-\Delta/2, \Delta/2]$. We assume that Z is available to both the encoder and the decoder. The ECDQ encoder encodes source X as $Q(X + Z)$. The quantized value is then described by a conditional universal entropy coder conditioned on Z. The decoder contains the corresponding conditional universal entropy decoder from which it decodes $Q(X + Z)$ and builds the reconstruction $\hat{X} = Q(X + Z) - Z$. We call this coding scheme Scalar ECDQ (SECDQ).

In a more general form of ECDQ, the uniform scalar quantizer is replaced by a K-dimensional lattice quantizer $Q : \mathcal{R}^K \to \mathcal{R}^K$ which maps every K-dimensional vector $x^K \in \mathcal{R}^K$ into the nearest lattice point l_i^K in the K-dimensional lattice L_K. Thus the set of all K-dimensional vectors mapped into $l_i^K \in L_K$ is the Voronoi region, $\mathcal{V}(l_i^K) = \{x^K \in \mathcal{R}^K : d(x^K - l_i^K) \leq d(x^K - l_j^K) \ \forall\, j \neq i\}$. The dither vector Z^K is a K-dimensional vector uniformly distributed over the basic cell of the lattice $\mathcal{V}_0 = \mathcal{V}(0^K)$, which is the Voronoi region of the lattice point 0^K. The ECDQ reconstructs X^K as $\hat{X}^K = Q(X^K + Z^K) - Z^K$. This coding scheme is called Lattice ECDQ (LECDQ).

In block coding, for a given n-dimensional vector $x^n \in \mathcal{R}^n$, we assume K divides n and consider x^n as a concatenation of n/K K-dimensional vectors. Each K-dimensional vector is quantized independently using identical dither; the concatenated n/K quantized values $Q^{n/K}(X^K + Z^K)$ are then coded jointly by conditional entropy codes.

We use the squared error distortion measure $d(x,\hat{x}) = (x-\hat{x})^2$. Let G_K denote the normalized second moment of L_K, then $E||Z^K||^2/K = G_K V^{2/K}$, where V is the volume of $\mathcal{V}_0$.

Properties of ECDQ [48]:

1. For reconstruction $\hat{X}^K = Q(X^K + Z^K) - Z^K$, the difference $\hat{X}^K - X^K$ is independent of X^K and is distributed as $-Z^K$. Thus the distortion

$$D = E||\hat{X}^K - X^K||^2/K = E||Z^K||^2/K = G_K V^{2/K}$$

is independent of the source X^K. If Z is a scalar ($K=1$) and uniformly distributed on $(-\Delta/2, \Delta/2]$, then $D = EZ^2 = \Delta^2/12$.

2. The rate of an ECDQ can be made arbitrarily close to the conditional entropy of the dithered quantizer given Z^K using block coding. Hence we measure the rate of the ECDQ as the conditional entropy of the quantizer output given Z^K. Thus the rate is

$$\frac{1}{K}H(Q(X^K + Z^K)|Z^K) = \frac{1}{K}I(X^K; X^K - Z^K).$$

3. Let the minimal G_K of any K-dimensional lattice be

$$G_K^{opt} = \min_Q (KV^{1+2/K})^{-1} \int_{V_0} ||x^K||^2 \mathrm{d}x^K.$$

Then for the corresponding optimal lattice quantizer, the autocorrelation of the quantizer noise is $R_{Z^K} = E(Z^K(Z^K)^t) = \sigma^2 I$, where I is the $K \times K$ identity matrix and $\sigma^2 = G_K^{opt} V^{2/K}$ is the second moment of the lattice.

Rate-distortion function

The rate-distortion function of a n-dimensional random vector X^n is defined as [47]

$$R_n(D) = \inf_{\{f_{U^n|X^n}(u^n|x^n):\ (1/n)Ed(X^n,U^n)\leq D\}} \frac{1}{n} I(X^n;U^n),$$

where $f_{U^n|X^n}(u^n|x^n)$ is a conditional probability density function (pdf) of the representation u^n given the source sample x^n, the term $(1/n)Ed(X^n,U^n)$ is the average distortion per source symbol between the source and its representation, and

$$I(X^n;U^n) = I(X_1,\ldots,X_n;U_1,\ldots,U_n)$$

is the mutual information between X^n and U^n.

Chapter Outline

In this chapter, I apply ECDQs to multi-resolution source coding and multiple access source coding.

Section 4.2 treats multi-resolution source codes. I show that using a nested scalar ECDQ in multi-resolution source coding can achieve a rate redundancy bounded from above by a constant for any distortion value at any resolution for any source. I also give a practical algorithm for designing multi-resolution scalar ECDQs and test that algorithm on a family of images.

Section 4.3 treats multiple access source codes. I apply lattice ECDQ and optimal linear estimation in multiple access source code design and derive the rate-distortion performance for arbitrary sources. A constant upper bound on the rate redundancy can be derived for some sources.

4.2 Multi-resolution Source Codes

A multi-resolution source code (MRSC) with $M = 2$ (2RSC) consists of two encoder maps and two decoder maps:

(a) a coarse map pair, encoder $f_1 : \mathcal{R}^n \to \{1, \ldots, L_1\}$ and decoder $g_1 : \{1, \ldots, L_1\} \to \mathcal{R}^n$ with rate $R_1 = (1/n) \log L_1$ and distortion $D_1 = (1/n) Ed(X^n, g_1(f_1(X^n)))$, and

(b) a refinement map pair, encoder $f_2 : \mathcal{R}^n \to \{1, \ldots, L_2\}$ and decoder $g_2 : \{1, \ldots, L_1\} \times \{1, \ldots, L_2\} \to \mathcal{R}^n$ with total rate $R_2 = (1/n) \log(L_1 L_2)$ and distortion $D_2 = (1/n)$ $Ed(X^n, g_2(f_1(X^n), f_2(X^n)))$.

The generalization to MRSC with $M > 2$ is straightforward; we use R_i and D_i, $i = 1, \ldots, M$, to denote the rate and distortion at the ith resolution.

The rate redundancy at the ith-resolution of a multi-resolution ECDQ is defined as the difference between the rate R_i achieved by the MR-ECDQ in resolution-i and the rate-distortion function $R_n(D_i)$ at the same distortion D_i.

4.2.1 Code Design and Performance Analysis

In this section, I consider scalar ECDQ in MRSC design. I use a nested coding structure and show that the rate redundancy of MR-SECDQ is bounded from above by a constant at all distortion values and all resolutions.

In the MR-SECDQ, I begin with a single dither random variable Z_1 distributed uniformly on $(-\Delta_1/2, \Delta_1/2]$ and then define $Z_i = Z_{i-1} - N_i \Delta_i$, $i = 2, 3, \ldots$, where $\Delta_i = \Delta_{i-1}/M_i$ for some positive integer M_i, and N_i is the nearest integer to Z_{i-1}/Δ_i. Since Z_1 is distributed

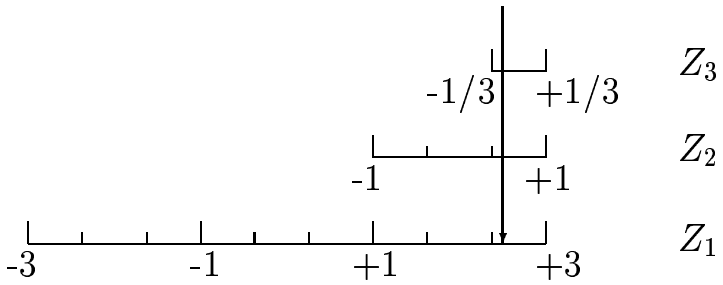


Figure 4.1: Calculating Z_2 and Z_3 as deterministic functions of dither random variable Z_1.

uniformly on $(-\Delta_1/2, \Delta_1/2]$, Z_i is uniformly distributed on interval $(-\Delta_i/2, \Delta_i/2]$. Figure 4.1 gives an example. Here $\Delta_1 = 6$, $M_1 = M_2 = M_3 = 3$, $Z_1 = 5/2$, hence $Z_2 = 1/2$ and $Z_3 = -1/6$.

The SECDQ encoder encodes source X by applying a uniform scalar quantizer Q_i to $X + Z_i$; the basic cell of Q_i is $(-\Delta_i/2, \Delta_i/2]$. The SECDQ describes the quantized value to the decoder using a conditional entropy coder conditioned on $Z_1, \ldots, Z_i, Q_1(X+Z_1), \ldots, Q_{i-1}(X+Z_{i-1})$. (Typically the entropy code is a block entropy code that describes every n quantized values using a conditional entropy coding on the block.) The decoder contains the corresponding universal conditional entropy decoder and outputs $\hat{X}_i = Q_i(X + Z_i) - Z_i$ as its reconstruction at resolution i.

For the nested MR-SECDQ, I have the following rate redundancy results.

Theorem 17 *Let $(D_1, \ldots, D_M)$ and $(R_1, \ldots, R_M)$ be the distortion and total rate at resolution $1, \ldots, M$ of a nested MR-SECDQ, where $D_i = D_{i-1}/M_i^2$ for some integer M_i. Then $R_i - R_n(D_i) \leq 0.754$ for all $i \in \{1, \ldots, M\}$ and any source.*

Proof. Assume we use block entropy coding with block length n.

By Property 1 from Section 4.1, the distortion at the i-th resolution is

$$D_i = \frac{1}{12}\Delta_i^2 = \frac{1}{12}\frac{1}{M_i^2}\Delta_{i-1}^2 = \frac{1}{M_i^2}D_{i-1} = \frac{1}{\prod_{j=2}^{i} M_j^2}D_1 = \frac{1}{12}\frac{1}{\prod_{j=2}^{i} M_j^2}\Delta_1^2.$$

Only nested distortion values $D_i = D_{i-1}/M_i^2$ can be obtained via the nested MR-SECDQ.

Let $Z_i^n = (Z_i, \ldots, Z_i)$ and use $Q_i^n(X^n + Z_i^n) = (Q_i(X_1 + Z_i), \ldots, Q_i(X_1 + Z_i))$ to denote a vector of n quantized values; then the rate at the i-th resolution is

$$\begin{aligned}
R_1 &= \frac{1}{n}H(Q_1^n(X^n + Z_1^n|Z_1)) = \frac{1}{n}I(X^n; X^n - Z_1^n) && (4.1)\\
R_i &= R_{i-1} + \frac{1}{n}H(Q_i^n(X^n + Z_i^n|Z_1, \ldots, Z_i, Q_1^n(X^n + Z_1^n), \ldots, Q_{i-1}^n(X^n + Z_{i-1}^n)))\\
&= R_{i-1} + \frac{1}{n}I(X^n; X^n - Z_i^n|X^n - Z_1^n, \ldots, X^n - Z_{i-1}^n) && (4.2)\\
&= \frac{1}{n}I(X^n; X^n - Z_i^n, X^n - Z_{i-1}^n, \ldots, X^n - Z_1^n)\\
&= \frac{1}{n}I(X^n; X^n - Z_i^n) + \frac{1}{n}I(X^n; X^n - Z_{i-1}^n, \ldots, X^n - Z_1^n|X^n - Z_i^n)\\
&= \frac{1}{n}I(X^n; X^n - Z_i^n) && (4.3)
\end{aligned}$$

where (4.1) and (4.2) come from Property 2 in Section 4.1, and (4.3) holds since Z_i uniquely determines Z_{i+1} in nested ECDQ.

Let U_i^n be the n-dimensional vector achieving the n-dimensional rate distortion function $R_n(D_i)$ for source X. Then applying [47, Theorem 2], I can bound the rate redundancy of the nested MR-SECDQ at the i-th resolution as

$$R_i - R_n(D_i) = \frac{1}{n}I(X^n; X^n - Z_i^n) - \frac{1}{n}I(X^n; U^n) \leq 0.754.$$

where $D_i = D_{i-1}/M_i^2$. □

For arbitrary distortion values at any resolution, I can obtain two upper bounds on the rate redundancy, one that allows time-sharing and one that doesn't. I consider the case without time-sharing first. The following result from [49] is useful in obtaining these results.

Lemma 12 *[49, Lemma 1] Let $R_n(D)$ be the rate-distortion function, then for any $0 < D_1 \leq D_2$,*

$$R_n(D_1) - R_n(D_2) \leq \frac{1}{2}\log\frac{D_2}{D_1}.$$

Theorem 18 *Let $(D_1, \ldots, D_M)$ and $(R_1, \ldots, R_M)$ be the target expected distortion and expected total rate at resolution $1, \ldots, M$ of an MR-SECDQ (without time-sharing). Then $R_i - R_n(D_i) \leq 1.754$ for all $i \in \{1, \ldots, M\}$ and any source.*

Proof. Given target distortions $D_1 \geq D_2 \geq \ldots \geq D_M$ at resolutions $1, \ldots, M$, we design a nested MR-SECDQ code with distortion $D_i' \leq D_i$ at each resolution i. We choose D_i' as $D_1' = D_1$, and, for each $i \in \{2, \ldots, M\}$, $D_i' = D_1/4^{m_i}$, where m_i is a positive integer satisfying $D_1/4^{m_i} \leq D_i < D_1/4^{m_i-1}$.

From Theorem 17, we can achieve $R_i - R_n(D_i') \leq 0.754$ for each $i \in \{1, \ldots, M\}$. Thus

$$\begin{aligned} R_i - R_n(D_i) &= R_i - R_n(D_i') + R_n(D_i') - R_n(D_i) \\ &\leq R_i - R_n(D_i') + \frac{1}{2}\log\frac{D_i}{D_i'} \qquad (4.4) \\ &\leq 0.754 + \frac{1}{2}\log 4 \qquad (4.5) \\ &= 1.754 \end{aligned}$$

where (4.4) comes from Lemma 12 and (4.5) follows from $D_i' = D_1/4^{m_i} \leq D_i < D_1/4^{m_i-1} = D_i'/4$. □

Lemma 13 is useful for the investigation of the time-sharing case.

Lemma 13 *Suppose $0 < D_2/4 \leq D_1 < D_2$ and $D = \alpha D_1 + (1-\alpha)D_2$, where $0 \leq \alpha \leq 1$, then*

$$\alpha R_n(D_1) + (1-\alpha)R_n(D_2) - R_n(D) < C_1 = 0.3413.$$

Proof:

$$\begin{aligned}
&\alpha R_n(D_1) + (1-\alpha)R_n(D_2) - R_n(D) \\
&= \alpha(R_n(D_1) - R_n(D)) + (1-\alpha)(R_n(D_2) - R_n(D)) \\
&\leq \alpha(R_n(D_1) - R_n(D)) && (4.6) \\
&\leq \frac{\alpha}{2}\log\frac{D}{D_1} && (4.7) \\
&= \frac{\alpha}{2}\log\frac{\alpha D_1 + (1-\alpha)D_2}{D_1} \\
&= \frac{\alpha}{2}\log\left(\alpha + (1-\alpha)\frac{D_2}{D_1}\right) \\
&\leq \frac{\alpha}{2}\log(4-3\alpha) && (4.8) \\
&\leq 0.3413, && (4.9)
\end{aligned}$$

where (4.6) follows since $D_2 \geq D$ and the rate-distortion function is a non-increasing function; (4.7) follows from Lemma 12; (4.8) follows from $D_2/4 \leq D_1$; and (4.9) comes from a standard maximization. □

Theorem 19 *Let $(D_1, \ldots, D_M)$ and $(R_1, \ldots, R_M)$ be the target expected distortion and expected total rate at resolution $1, \ldots, M$ of an MR-SECDQ that allows time-sharing. Then $R_i - R_n(D_i) \leq 1.096$ for all $i \in \{1, \ldots, M\}$ and any source.*

Proof. I use time sharing between the resolutions of a nested MR-SECDQ. The nested MR-SECDQ achieves rate R'_j and distortion D'_j at resolution j, where $D'_1 = D_1$ and

$$D'_j = \frac{1}{4^{m_i-1}}D_1 > D_i \geq \frac{1}{4^{m_i}}D_1 = D'_{j+1} \quad \text{for } j = 2i,\ i > 1.$$

I achieve distortion D_i in resolution i of the target MR-SECDQ by time-sharing between the $(2i)$-th and the $(2i+1)$-th resolution of the nested MR-SECDQ. Since $D'_{2i} > D_i \geq D'_{2i+1}$,

there must exist an $\alpha \in [0,1]$ such that $\alpha D'_{2i+1} + (1-\alpha)D'_{2i} = D_i$. Hence $R_i = \alpha R'_{2i+1} + (1-\alpha)R'_{2i}$. The rate redundancy can be bounded by $R_1 - R_n(D_1) = R'_1 - R_n(D'_1) \leq 0.754$ and for $i > 1$,

$$\begin{aligned}
& R_i - R_n(D_i) \\
&= \alpha R'_{2i+1} + (1-\alpha)R'_{2i} - R_n(\alpha D'_{2i+1} + (1-\alpha)D'_{2i}) \\
&= \alpha(R'_{2i+1} - R_n(D'_{2i+1})) + (1-\alpha)(R'_{2i} - R_n(D'_{2i})) \\
&\quad + \alpha R_n(D'_{2i+1}) + (1-\alpha)R_n(D'_{2i}) - R_n(\alpha D'_{2i+1} + (1-\alpha)D'_{2i}) \\
&\leq 0.754 + \alpha R_n(D'_{2i+1}) + (1-\alpha)R_n(D'_{2i}) - R_n(\alpha D'_{2i+1} + (1-\alpha)D'_{2i}) \qquad (4.10) \\
&\leq 0.754 + 0.3413 \qquad (4.11) \\
&= 1.096,
\end{aligned}$$

where (4.10) comes from Theorem 17, and (4.11) follows from Lemma 13. □

In the above argument, I time-share between the resolutions of a single MR-SECDQ code. To do this, for a fraction α of the time, I use the j-th resolution by encoding, transmitting and decoding $Q_1, \ldots, Q_j$; for the remaining fraction $(1-\alpha)$ of the time, I use the $(j+1)$-th resolution by encoding, transmitting and decoding $Q_1, \ldots, Q_j, Q_{j+1}$. The resulting rate redundancy bound is a constant independent of the distortion values, the number of resolutions, and the source. If we allow the rate redundancy bound to be expressed as a function of the distortion values, we can time-share among distinct nested MR-SECDQ codes and achieve a tighter upper bound.

4.2.2 Code Implementation and Experimental Results

Implementing the MR-SECDQs described in the previous section presents certain difficulties, since universal conditional entropy codes are not currently available and even good low-complexity conditional entropy codes without the universality constraint are difficult to achieve. In order to implement a simple coding algorithm that takes advantage of the correlation among adjacent symbols and uses information from previous resolutions and the dithers to improve the performance, I modify the code design techniques of Section 4.2.1 and give a practical multi-resolution source coding algorithm.

Code Implementation

Consider a length-L data sequence $\{X_i\}_{i=1}^{L}$, where each $X_i \in [X_{\min}, X_{\max}]$. Figure 4.2(a) gives a sample image file. The data sequence is obtained in line scan order, i.e., by reading from left to right, from top to bottom of the image. I fix the value of the dither $Z \equiv z_o$, $z_o \in [-X_{\max}, X_{\max}]$, and add it to every symbol in the sequence (see Figure 4.2(a)).[1]

At the first resolution, quantize $X_i + z_o$ using uniform scalar quantizer $Q_1(.)$ with basic cell size $\Delta_1 = X_{\max} - X_{\min}$. The index of the quantized value is $I_1(X) = \lfloor X/\Delta_1 \rfloor$. We use universal entropy code (e.g., Lempel-Ziv-Welch code [50, 51] or Burrows-Wheeler Transform coding [52]) on $I_1(X_1 + z_o)I_1(X_2 + z_o) \ldots I_1(X_L + z_o)$. The decoder decodes $I_1(X_i + z_o)$

[1] In Theorems 17,18 and 19 of Section 4.2.1, the rate-distortion performance we obtained is an expected performance over the distribution of source X and dither Z. So there always exists a value $z^\star$ of Z that achieves rate-distortion performance above average and is at most a constant distance away from the optimum. I fix the value of Z for each image. Through experiments, I develop a rule for determining a good value of Z, as discussed shortly.

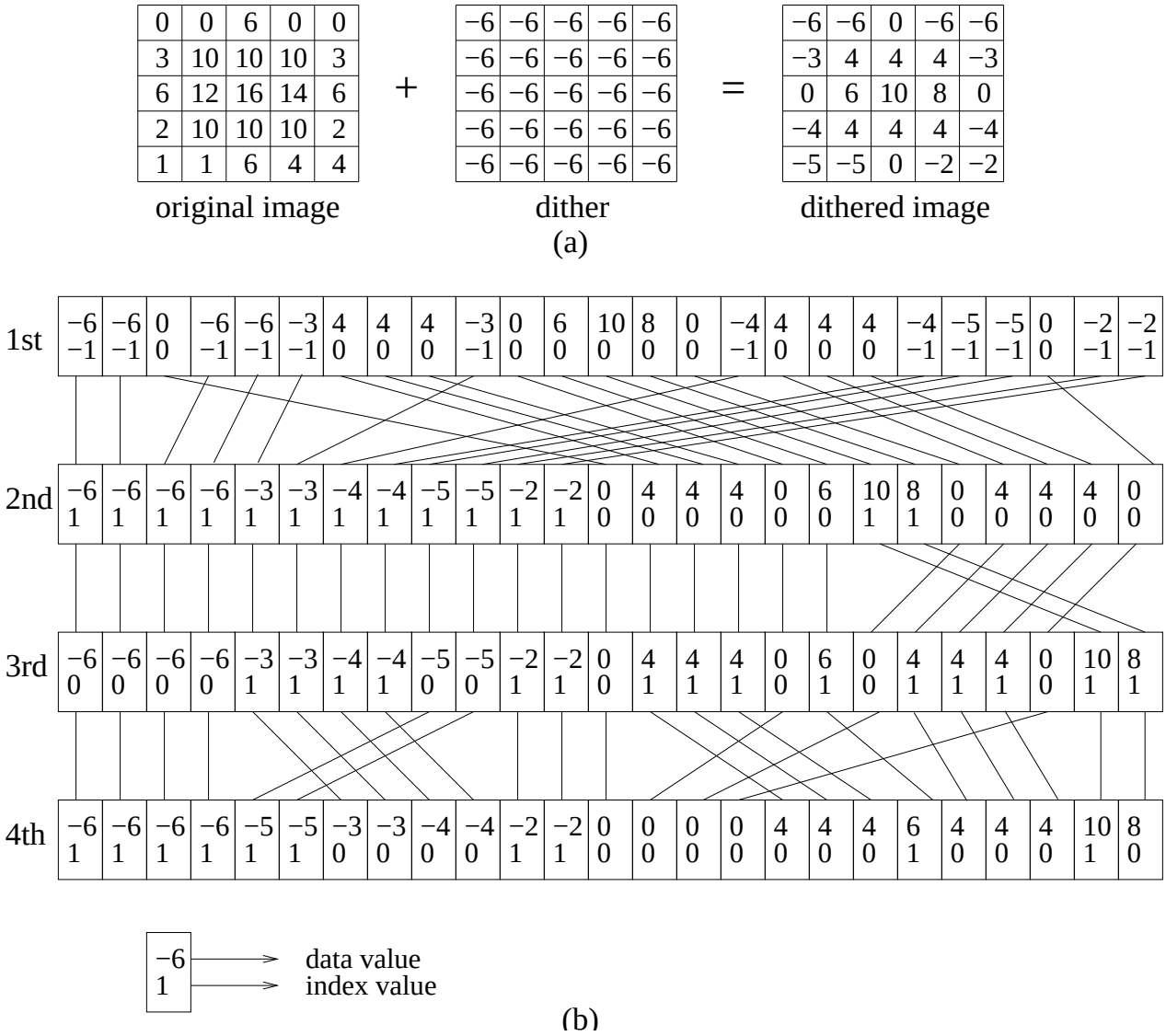


Figure 4.2: MR-ECDQ implementation: (a) original image and dithered image; (b) rearranging pixels positions in multi-resolution coding.

and reconstructs $Q_1(X_i + z_o)$ as the mid-point of the quantizing cell, i.e., $Q_1(X_i + z_o) = I_1(X_i+z_o)\Delta_1+\Delta_1/2$. The reconstruction for X_i at the 1st resolution is $\hat{X}_{1,i} = Q_1(X_i+z_o)-z_o$, $i = 1, \ldots, L$.

At the m-th ($m > 1$) resolution, I first re-order the symbols in the sequence, so that symbols with identical index values $I_1, I_2, \ldots, I_{m-1}$ are placed at adjacent positions (see Figure 4.2 for an illustration). The index value $I_m(X_i + z_o)$ conditioned on $Q_{m-1}(X_i + z_o)$ is

$$I_m(X_i + z_o) = \begin{cases} 0 & \text{if } X_i + z_o < Q_{m-1}(X_i + z_o), \\ 1 & \text{if } X_i + z_o \geq Q_{m-1}(X_i + z_o). \end{cases}$$

The binary sequence $I_m(X_{m(1)}+z_o)I_m(X_{m(2)}+z_o)\ldots I_m(X_{m(L)}+z_o)$ is then coded by universal entropy code, where $X_{m(1)}, \ldots, X_{m(L)}$is the data sequence after re-ordering. The decoder already knows I_k and Q_k for all X and all $k < m$; hence the decoder can recover $I_m(X + z_o)$ and re-order the data sequence appropriately. The reproduction value is the mid-point of the new quantization cell: $Q_m(X_i + z_o) = Q_{m-1}(X_i + z_o) + I_m(X_i + z_o)\Delta_m - \Delta_m/2$. The reconstruction for X_i at the m-th resolution is $\hat{X}_{m,i} = Q_m(X_i + z_o) - z_o$, $i = 1, \ldots, L$.

The above procedure is equivalent to quantize $X_i + z_o$ using uniform scalar quantizer $Q_m(.)$ with basic cell size $\Delta_m = \Delta_{m-1}/2$, then code $Q_m(X_i + z_o)$ using a universal conditional entropy code conditioned on $Q_1(X_i + z_o), \ldots, Q_{m-1}(X_i + z_o)$. By rearranging the positions of pixels according to $I_1, \ldots, I_{m-1}$ (or equivalently $Q_1(X_i+z_o), \ldots, Q_{m-1}(X_i+z_o)$), a simple universal entropy code can capture the statistics of $Q_m(X_i + z_o)$ given $Q_1(X_i + z_o), \ldots, Q_{m-1}(X_i + z_o)$ and achieve performance close to the optimal performance.

There are a few techniques that can further improve the performance and still keep the algorithm simple.

1. Scan the image in a zigzag pattern as shown in Figure 4.3.

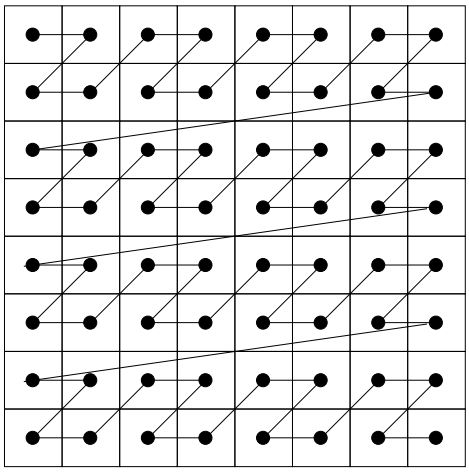


Figure 4.3: The zigzag scanning pattern.

2. Set $Q_m(\cdot)$ as the centroid (rather than the mid-point) of the quantization cell and use non-uniform scalar quantizer for the m-th ($m \geq 1$) resolution.

3. Experimentally I find $z_o = (X_{\max} - X_{\min})/2 - \sum_{i=1}^{L} X_i/L$ to be a good value for the dither.

Experimental Results

I implement the algorithm of previous section and test it on a variety of images. I use both the Lempel-Ziv-Welch (LZW) code and BZIP, an algorithm based on Burrows-Wheeler Transform (BWT) as lossless codes. I compare the rate-distortion performance of variations on the above algorithm with each other and with the performance of the Set Partitioning In Hierarchical Trees (SPIHT) algorithm [53, 54]. The images I test include Barbara, Crowd, Airport, Lena, and a set of medical brain-scan images (shown in the Appendix).

First, I test the influence of the quantization rule, entropy code, and image scan order. Figure 4.4 shows the average results on a brain-scan image Brain1 averaged over a range of

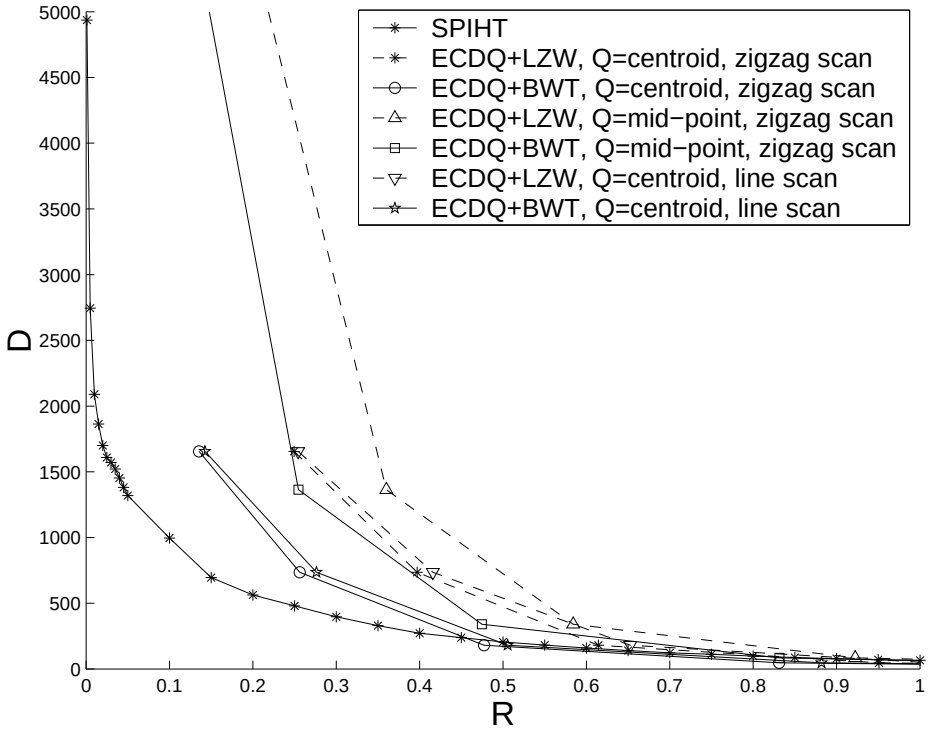
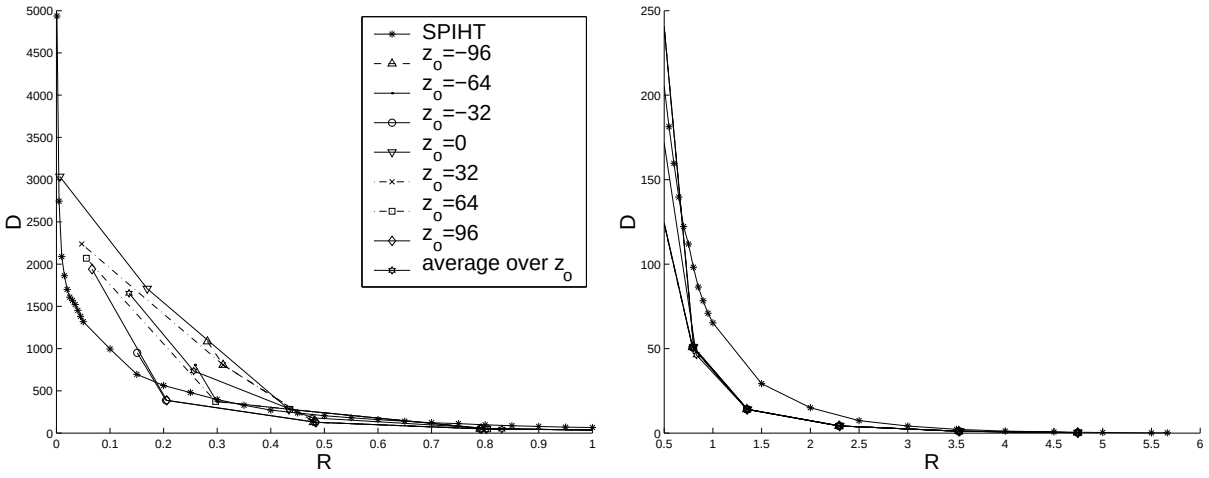


Figure 4.4: Various methods for improving the performance of ECDQ.

Z values. The results are similar for other images. Here centroids outperform mid-points (despite the overhead needed to describe those centroids to the decoder), BTW outperforms LZW, and zigzag outperforms raster scan.

In Figure 4.5, I plot the rate-distortion curves for different dither values for image Brain1 using BWT in ECDQ. The dither value z_o has much greater influence on low rate performance than on high rate performance. For $R < 0.2$, $z_o = 96$ achieves the best performance over all tested dither values and its rate performance differs from that of SPIHT by no more than 0.08 bits/symbol at the same distortion. For $R \geq 0.5$, most z_o values outperform SPIHT, in some cases yielding rate differences larger than 0.5 for the same distortion.

Figure 4.6 compares the performance achieved by applying ECDQ on the original image to that achieved by applying ECDQ on the image's wavelet coefficients [53]. The latter



(a) At low resolution.

(b) At high resolution.

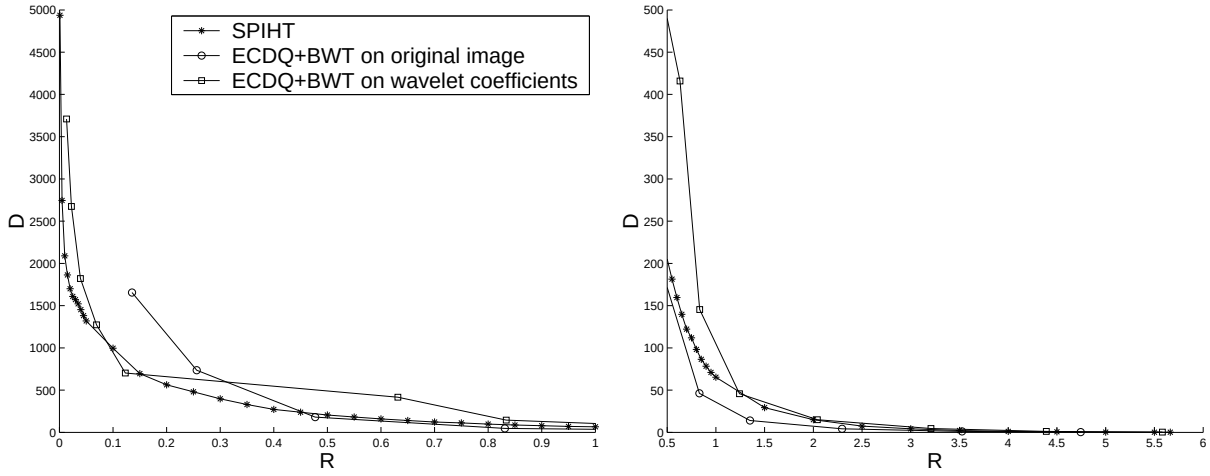
Figure 4.5: ECDQ+BWT's performance on a brain-scan image at different z_o values.

achieves performance very close to that of SPIHT at very low rates (the rate difference is at most 0.02). At higher rates, application of ECDQ in the spatial domain outperforms both application of ECDQ to the wavelet coefficients and SPIHT.

In Figures 4.7, 4.8, 4.9 and 4.10, I compare the performance of ECDQ and SPIHT on four images. Our conclusions from these figures are: at very low rates, the performance of ECDQ on wavelet coefficients can be very close to or even the same as that of SPIHT; at higher rates ECDQ on the original image always outperforms SPIHT.

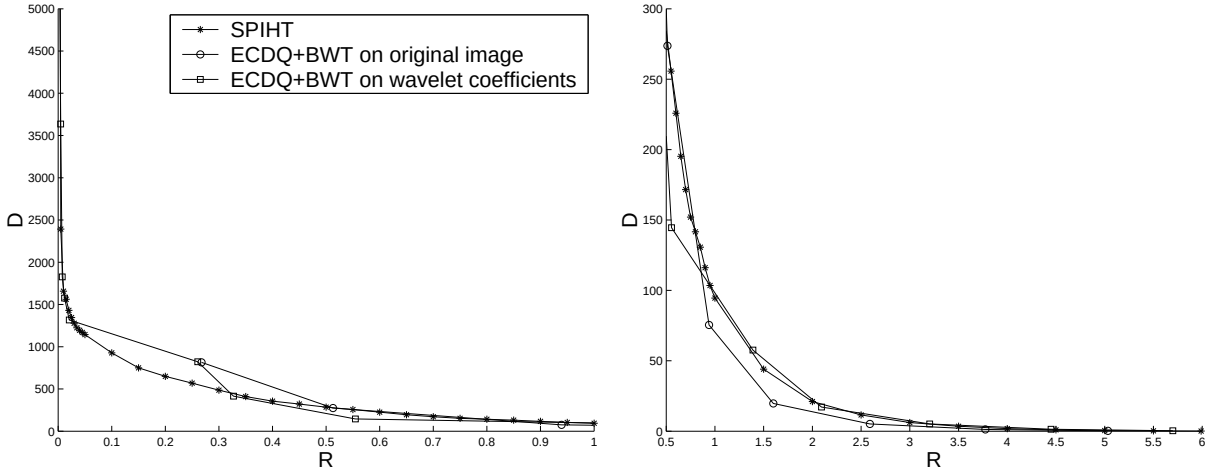
4.3 Multiple Access Source Codes

A multiple access source code (MASC) with $M = 2$ (2ASC) consists of two encoder maps, $f_1 : \mathcal{R}^n \to \{1, \ldots, L_1\}$ and $f_2 : \mathcal{R}^n \to \{1, \ldots, L_2\}$, and one decoder map, $g : \{1, \ldots, L_1\} \times \{1, \ldots, L_2\} \to \mathcal{R}^n \times \mathcal{R}^n$. The rates and distortions are $R_i = (1/n) \log L_i$, $i = 1, 2$, and $D_1 = (1/n)Ed(X^n, g_1(f_1(X^n), f_2(Y^n)))$, $D_2 = (1/n)Ed(Y^n, g_2(f_1(X^n), f_2(Y^n)))$, respectively.



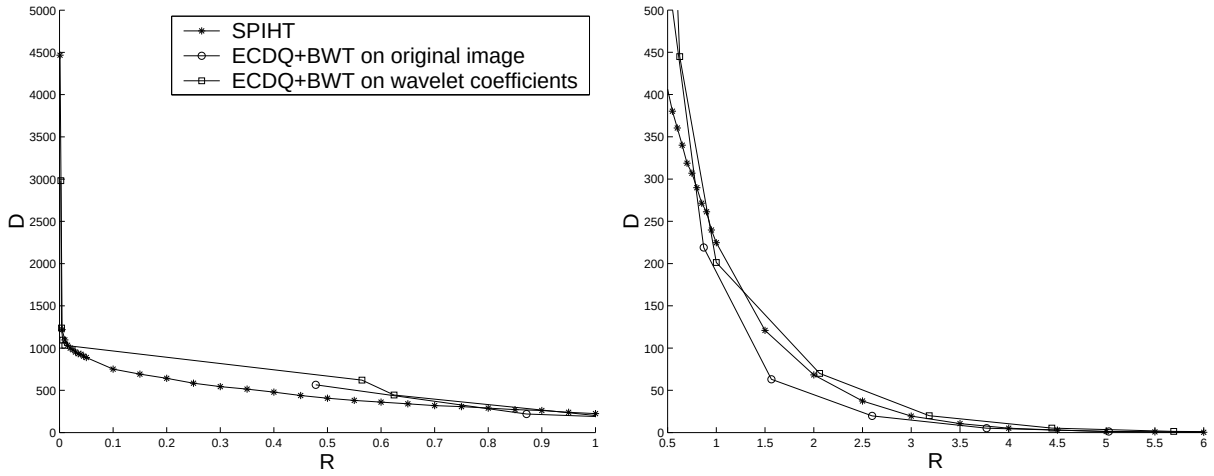
(a) At low resolution. (b) At high resolution.

Figure 4.6: Comparison of the average performance of different ECDQ algorithms with SPIHT on a brain-scan image Brain1.



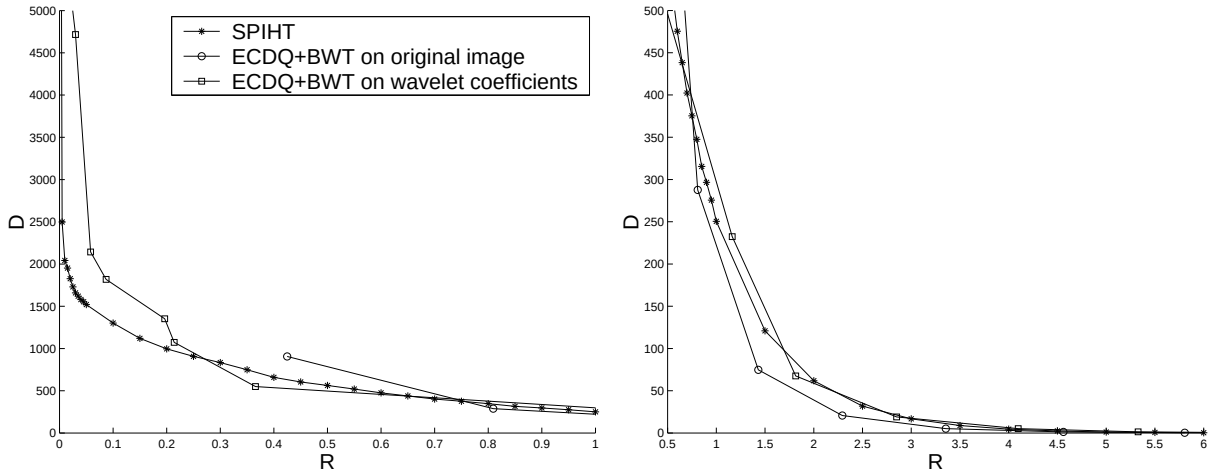
(a) At low resolution. (b) At high resolution.

Figure 4.7: Comparison of the performance of ECDQ with SPIHT on image Lena.



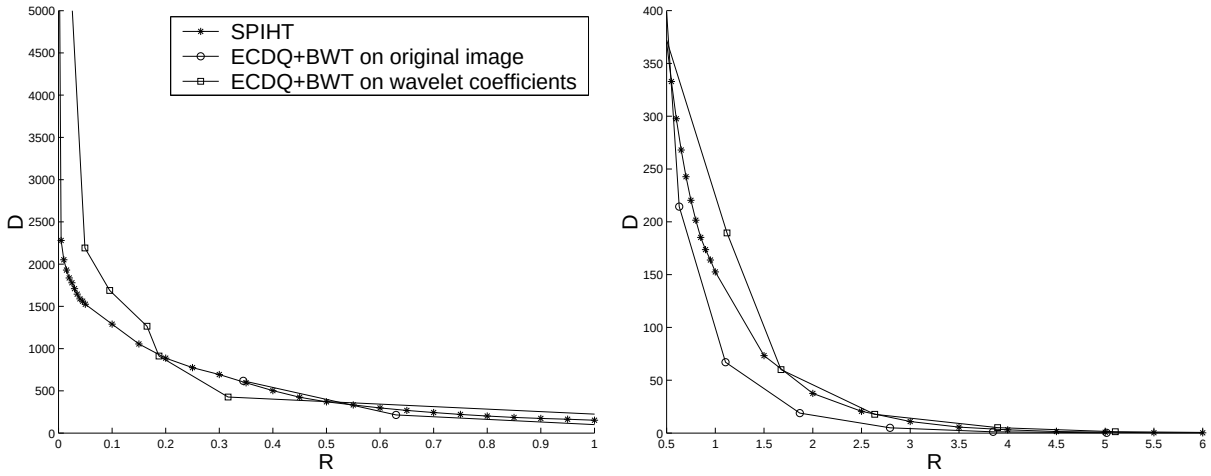
(a) At low resolution. (b) At high resolution.

Figure 4.8: Comparison of the performance of ECDQ with SPIHT on image Airport.



(a) At low resolution. (b) At high resolution.

Figure 4.9: Comparison of the performance of ECDQ with SPIHT on image Barbara.



(a) At low resolution. (b) At high resolution.

Figure 4.10: Comparison of the performance of ECDQ with SPIHT on image Crowd.

Tung and Berger investigate the lossy source coding problem for this system and give one inner bound and one outer bound on the achievable region in [55]. The inner bound is not tight in general, and the relationship between the inner bound and the achievable region for joint encoding is difficult to understand. Feng and Effros bound the difference between the performance of MASCs and the joint rate-distortion function in [56].

To apply K-dimensional lattice ECDQ in MASC design (MA-ECDQ), I assume $K = n$ in the following discussion. The results apply to any K such that K divides n.

Let X^n and Y^n be n-dimensional vector sources. I use independent ECDQs on X^n and Y^n and use joint optimal linear estimation at the decoder to get reconstructions $\hat{X}^n$ and $\hat{Y}^n$. Let Z_1^n and Z_2^n be the n-dimensional dithers with $Z_1^n \perp\!\!\!\perp Z_2^n$, $(Z_1^n, Z_2^n) \perp\!\!\!\perp (X^n, Y^n)$, $\sigma_i^2 I = 1/nE(Z_i^n (Z_i^n)^t)$, $i = 1, 2$, where $A \perp\!\!\!\perp B$ specifies that A and B are independent. Then the channel codewords c_1 of X^n and c_2 of Y^n are the entropy coding of $Q_1(X^n + Z_1^n)$ and

$Q_2(Y^n + Z_2^n)$ conditioned on Z_1^n and Z_2^n, respectively. The reconstruction of X^n is

$$\hat{X}^n = \alpha_1(Q_1(X^n + Z_1^n) - Z_1^n) + \beta_1(Q_2(Y^n + Z_2^n) - Z_2^n)$$

and that of Y^n is

$$\hat{Y}^n = \alpha_2(Q_1(X^n + Z_1^n) - Z_1^n) + \beta_2(Q_2(Y^n + Z_2^n) - Z_2^n),$$

for some coefficients $\alpha_1, \alpha_2, \beta_1, \beta_2$. Let $\sigma_X^2 = (1/n)\sum_{i=1}^n E(X_i^2)$, $\sigma_Y^2 = (1/n)\sum_{i=1}^n E(Y_i^2)$, and $(1/n)\sum_{i=1}^n E(X_iY_i) = \rho\sigma_X\sigma_Y$. Then the distortion for X^n is

$$\begin{aligned}
& D_1 \\
= \; & \frac{1}{n}E||X^n - \hat{X}^n||^2 \\
= \; & \frac{1}{n}E||X^n - (\alpha_1(Q_1(X^n + Z_1^n) - Z_1^n) + \beta_1(Q_2(Y^n + Z_2^n) - Z_2^n))||^2 \\
= \; & \frac{1}{n}E||((1-\alpha_1)X^n - \beta_1 Y^n) + \alpha_1(X^n + Z_1^n - Q_1(X^n + Z_1^n)) + \beta_1(Y^n + Z_2^n - Q_2(Y^n + Z_2^n))||^2 \\
= \; & (1-\alpha_1)^2\sigma_X^2 - 2\beta_1(1-\alpha_1)\rho\sigma_X\sigma_Y + \beta_1^2\sigma_Y^2 + \alpha_1^2\sigma_1^2 + \beta_1^2\sigma_2^2.
\end{aligned}$$

We want to minimize D_1 for fixed σ_1^2 and σ_2^2. Let $\partial D_1/\partial\alpha_1 = 0$ and $\partial D_1/\partial\beta_1 = 0$, then

$$\alpha_1(\sigma_X^2 + \sigma_1^2) + \beta_1\rho\sigma_X\sigma_Y = \sigma_X^2, \quad \alpha_1\rho\sigma_X\sigma_Y + \beta_1(\sigma_Y^2 + \sigma_2^2) = \rho\sigma_X\sigma_Y.$$

Thus,

$$\alpha_1 = \frac{(1-\rho^2)\sigma_X^2\sigma_Y^2 + \sigma_X^2\sigma_2^2}{\Lambda}, \quad \text{and} \quad \beta_1 = \frac{\rho\sigma_X\sigma_Y\sigma_1^2}{\Lambda},$$

where $\Lambda = (\sigma_X^2 + \sigma_1^2)(\sigma_Y^2 + \sigma_2^2) - \rho^2\sigma_X^2\sigma_Y^2$. As a result,

$$D_1 = \frac{\sigma_X^2\sigma_1^2}{\Lambda^2}\left[(\sigma_Y^2 + \sigma_2^2)(\sigma_X^2 + \sigma_1^2) - \rho^2\sigma_X^2\sigma_Y^2\right]\left[\sigma_Y^2 + \sigma_2^2 - \rho^2\sigma_Y^2\right] = \frac{\sigma_X^2\sigma_1^2((1-\rho^2)\sigma_Y^2 + \sigma_2^2)}{\Lambda}.$$

By symmetry,

$$D_2 = \frac{\sigma_Y^2\sigma_2^2((1-\rho^2)\sigma_X^2 + \sigma_1^2)}{\Lambda}.$$

The rates are

$$R_1 = \frac{1}{n} I(X^n; X^n - Z_1^n) \quad \text{and} \quad R_2 = \frac{1}{n} I(Y^n; Y^n - Z_2^n).$$

I examine the performance of this code in a simple case. When X^n and Y^n are iid and joint Gaussian, $\sigma_X^2 = \sigma_Y^2 = 1$, and $D_1 = D_2 = D$, the joint rate-distortion function is [3]:

$$R_{X^n Y^n}(D, D) = \begin{cases} \frac{1}{2} \log \frac{1-\rho^2}{D^2}, & 0 \le D \le 1-\rho, \\ \frac{1}{2} \log \frac{1+\rho}{2D-(1-\rho)}, & 1-\rho \le D \le 1, \\ 0, & D \ge 1. \end{cases}$$

We have

$$R_1 \le \frac{1}{2} \log \frac{1+\sigma_1^2}{\sigma_1^2} + \frac{1}{2} \log(2\pi e G_n).$$

Since $D_1 = D_2$, we get $\sigma_1^2 = \sigma_2^2$ and $R_1 = R_2$. Consider $n \to \infty$, then $\sigma_1^2 = 1/(2^{2R_1} - 1)$, hence

$$D_1 = \frac{\sigma_X^2 \sigma_1^2 ((1-\rho^2)\sigma_Y^2 + \sigma_2^2)}{\Lambda} \le \frac{(2^{2R_1} - 1)(1-\rho^2) + 1}{2^{4R_1} - \rho^2 (2^{2R_1} - 1)^2}.$$

Figure 4.11 illustrates the comparison of the upper bound on the total rates $R_1 + R_2$ achieved by MA-ECDQ to the joint rate-distortion function $R_{X^n Y^n}$, where we assume that $n \to \infty$, $\sigma_X^2 = \sigma_Y^2 = 1$ and $\rho = 1/2$. This figure shows that the maximum difference does not exceed 0.2075.

4.4 Summary

In this chapter, I design multi-resolution source codes and multiple access source codes using entropy constrained dithered quantization. In both cases, I analyze the rate-distortion performance. For multi-resolution source coding, I use a nested scalar ECDQ and demonstrate

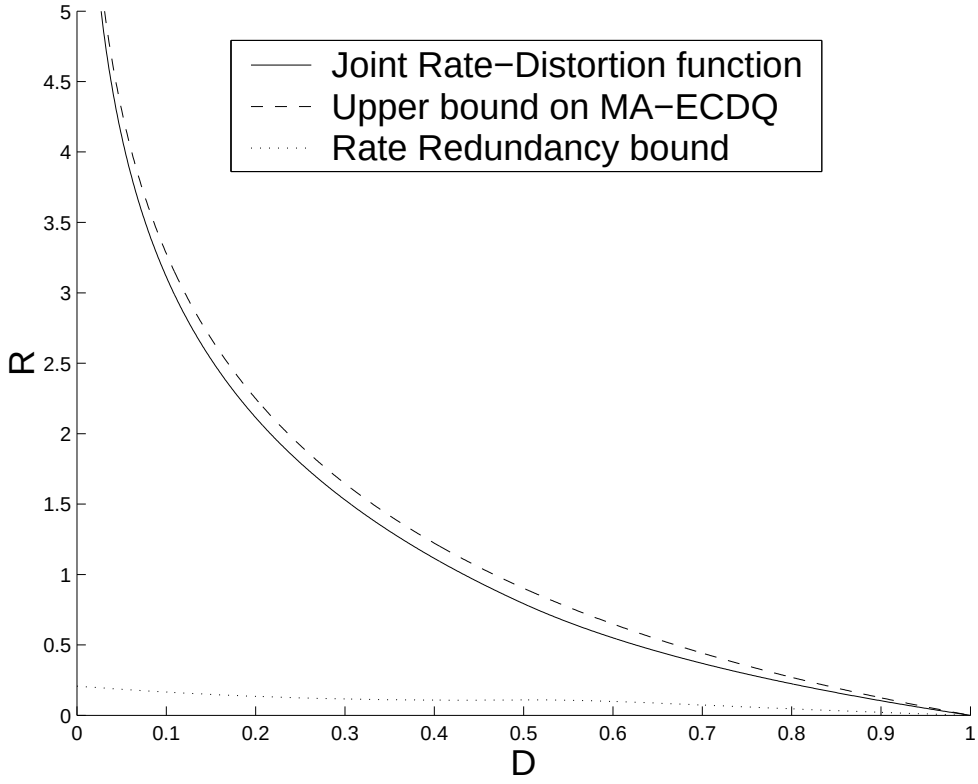


Figure 4.11: Comparing the rate achieved by MA-ECDQ with the minimal achievable rate at the same distortion for Gaussian sources.

constant rate redundancy bound at all resolutions, for all distortions, and for all sources. I further provide a practical implementation of MR-SECDQ and show its competitive rate-distortion performance on images. For multiple access source coding, I provide a lossy coding scheme using lattice ECDQ combined with optimal linear estimation at the decoder. I show that the code can achieve rate-distortion performance very close to the joint rate-distortion function for certain Gaussian sources.

Chapter 5

Summary

In this thesis, I present a variety of source coding techniques for use in network environments and demonstrate the concrete benefits of network source codes over traditional source codes from both theoretical and practical perspectives.

The network systems I consider include broadcast systems, multiple access systems, and multi-resolution systems. I investigate various code properties, analyze limiting performance, and design optimal and low-complexity coding algorithms for lossless, near-lossless, and lossy source coding in those systems. The experiments I performed on various data sets demonstrate the power of these algorithms.

In particular, for broadcast systems, I prove the theoretical limits of lossless source coding performance and use a tree-structured vector quantizer for lossy code design. The tree-structured vector quantizer design can also be applied to multiple description and multi-resolution source coding which are special cases of broadcast system source coding.

For multiple access systems, I first address instantaneous code and break optimal lossless and near-lossless code design into partition design and matched code design. I describe

the properties of optimal partitions and optimal matched code. These properties relates network source coding with traditional (single-sender, single-receiver) source coding and can decrease the complexity associated with optimal multiple access source code design. These results yield a means of searching for the optimal code for an arbitrary source p.m.f. $p(x, y)$. Since optimal MASC design is NP-hard, I provide a family of low-complexity sub-optimal algorithms which approximate the optimal design for general p.m.f.s. Both the optimal and sub-optimal MASC design algorithm has been applied in various network source coding scenarios. I also investigate a larger class of code: the uniquely decodable MASC. I provide necessary and sufficient conditions on the actual codewords and necessary conditions on the codeword lengths for uniquely decodable MASCs. Lower bounds on the MASC's achievable rates are also given. Experimental results are consistent with theory.

In seeking for a simple lossy code design algorithm in network environment, I turn to entropy constrained dithered quantization and apply ECDQ to multi-resolution and multiple access systems. I analyze the rate-distortion performance for each code and show ECDQ applied in networks can achieve performance very close to the theoretical optimal performance. I also provide a practical implementation of ECDQ in MR systems and show its competitive rate-distortion performance on images.

The analysis generalize and I apply traditional source coding techniques to network coding environments. These techniques include entropy coding (e.g., Huffman coding, arithmetic coding, and the Kraft inequality), universal coding (e.g., Lempel-Ziv coding, Burrow-Wheeler Transform coding), quantization (e.g., weighted universal vector quantization, tree-structured vector quantization, scalar or lattice entropy constrained dithered quantization), Lagrangian optimization and rate-distortion theory. The theory of simulated annealing is

also used in giving low complexity design algorithms.

Chapter 6

Appendix

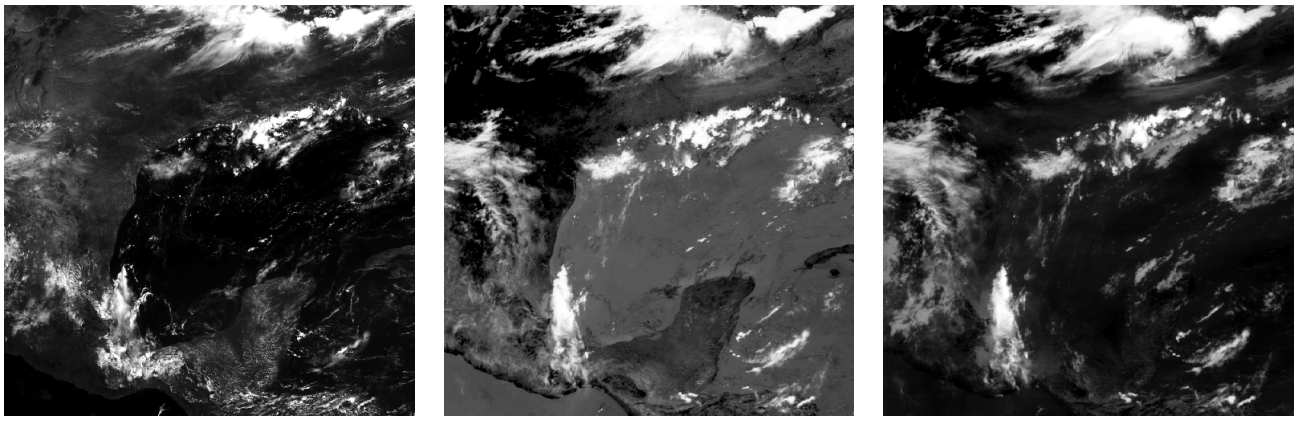
Derivation of $r_1[i,j,k]$ in Theorem 10

Recall that in arithmetic coding, the optimal description length of node $\mathbf{n}k$ is $l(\mathbf{n}k) = l(\mathbf{n}) - \log \frac{Q(\mathbf{n}k)}{\sum_{j=1}^{K(\mathbf{n})} Q(\mathbf{n}j)}$. Suppose $\mathcal{G}[i,j] = [\mathcal{G}_1, \ldots, \mathcal{G}_{N_1}]$, $\mathcal{G}[j+1,k] = [\mathcal{G}'_1, \ldots, \mathcal{G}'_{N_2}]$, let $r(\mathcal{G})$ be the best rate of $\mathcal{G}$, $Q(\mathcal{G})$ be the subtree probability of $\mathcal{G}$'s root, then

$$\begin{aligned} r[i,j] &= \sum_{i=1}^{N_1} r(\mathcal{G}_i) + \sum_{i=1}^{N_1} -Q(\mathcal{G}_i) \log \frac{Q(\mathcal{G}_i)}{P[i,j]} \\ &= \sum_{i=1}^{N_1} r(\mathcal{G}_i) + \sum_{i=1}^{N_1} -Q(\mathcal{G}_i) \log Q(\mathcal{G}_i) + P[i,j] \log P[i,j] \\ r[j+1,k] &= \sum_{i=1}^{N_2} r(\mathcal{G}'_i) + \sum_{i=1}^{N_2} -Q(\mathcal{G}'_i) \log Q(\mathcal{G}'_i) + P[j+1,k] \log P[j+1,k]. \end{aligned}$$

The new group $\mathcal{G}[i,j]$ is obtained by putting $\mathcal{G}[i,j]$ and $\mathcal{G}[j+1,k]$ as siblings, thus

$$\begin{aligned} & r_1[i,j,k] \\ &= \sum_{i=1}^{N_1} r(\mathcal{G}_i) + \sum_{i=1}^{N_2} r(\mathcal{G}'_i) + \sum_{i=1}^{N_1} -Q(\mathcal{G}_i) \log Q(\mathcal{G}_i) + \sum_{i=1}^{N_2} -Q(\mathcal{G}'_i) \log Q(\mathcal{G}'_i) + P[i,k] \log P[i,k] \\ &= r[i,j] + r[j+1,k] - P[i,j] \log P[i,j] - P[j+1,k] \log P[j+1,k] + P[i,k] \log P[i,k] \\ &= r[i,j] + r[j+1,k] + P[i,k] H(\frac{P[i,j]}{P[i,k]}). \end{aligned}$$



(a) Visible Spectrum (b) Infrared 2 (c) Infrared 5

Figure 6.1: Sample images from the GOES-8 weather satellites, used in BSVQ.

Joint Probability Examples Used in Chapter 3

Tables 6.1 and 6.2 give four 8×8 and two 16×16 p.m.f. examples.

Test Image Examples

This section gives a few examples of the test images I use in this thesis.

Table 6.1: Sample p.m.f.s on alphabet $\mathcal{X} \times \mathcal{Y}$ with $\mathcal{X} = \mathcal{Y} = \{a_0, a_1, \ldots, a_6, a_7\}$.

(a)

x\ y	a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7
a_0	.04	0	.04	.02	0	0	0	0
a_1	0	.04	0	0	.05	.1	0	0
a_2	.15	0	.05	0	0	0	0	0
a_3	0	.05	0	.06	0	0	0	0
a_4	0	.06	0	0	.05	0	0	0
a_5	0	0	0	.01	.02	.03	0	0
a_6	0	0	.01	0	0	.06	.02	.01
a_7	0	0	0	0	0	0	.05	.08

(b)

x\ y	a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7
a_0	.1	0	.1	.1	0	0	0	0
a_1	.06	.04	0	0	.05	0	.1	0
a_2	.15	0	.05	0	0	0	0	0
a_3	0	.05	0	.05	0	0	0	0
a_4	0	.04	0	0	.02	0	0	0
a_5	0	0	0	.02	.01	.01	0	0
a_6	0	0	.01	0	0	0	.015	.005
a_7	0	0	0	0	0	.01	0	.01

(c)

x\ y	a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7
a_0	.06	0	0	0	.04	0	0	0
a_1	0	0	.05	.04	0	0	0	0
a_2	0	0	.05	0	0	0	0	.15
a_3	.06	0	0	0	.05	0	0	0
a_4	.05	0	0	0	0	.06	0	0
a_5	0	0	0	0	0	0	.07	0
a_6	0	.06	0	.1	0	0	0	.03
a_7	0	0	0	0	.08	0	.05	0

(d)

x\ y	a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7
a_0	.06	.04	0	0	0	0	0	0
a_1	0	.04	.05	0	0	0	0	0
a_2	0	0	.05	.15	0	0	0	0
a_3	0	0	0	.06	.05	0	0	0
a_4	0	0	0	0	.05	.06	0	0
a_5	0	0	0	0	0	0	.07	0
a_6	0	0	0	0	0	.1	.06	.03
a_7	0	0	0	0	0	0	.05	.08

Table 6.2: Sample p.m.f.s $p(x, y)$ on alphabet $\mathcal{X} \times \mathcal{Y}$ with $\mathcal{X} = \mathcal{Y} = \{a_0, a_1, \ldots, a_{15}\}$.

(a) $356 \cdot p(x, y)$

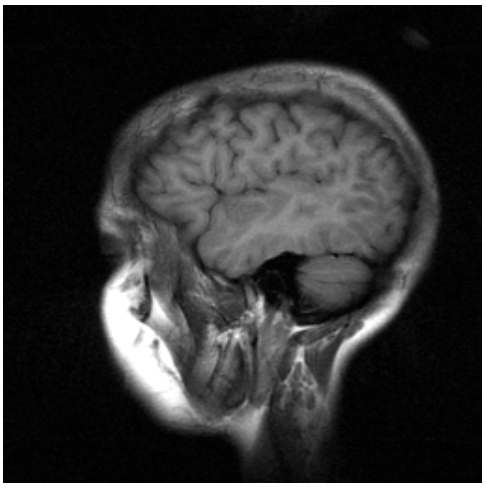
x\ y	a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}
a_0	7	3	0	0	6	4	0	0	3	2	0	0	0	0	0	0
a_1	0	9	0	0	4	3	0	0	0	3	0	0	0	0	0	0
a_2	0	6	0	0	0	6	0	0	0	6	0	0	0	0	0	0
a_3	0	0	3	2	0	0	7	3	0	0	3	3	0	0	0	0
a_4	1	0	2	0	0	0	1	1	0	0	2	0	1	1	0	0
a_5	3	3	0	6	3	3	0	0	3	3	0	6	3	3	0	0
a_6	0	0	6	0	0	0	9	6	0	0	6	0	0	0	6	6
a_7	3	3	0	3	3	3	0	0	3	3	0	3	3	3	0	0
a_8	4	0	2	0	2	0	2	1	2	0	2	0	2	0	2	0
a_9	0	2	0	1	0	2	0	0	0	2	0	1	0	2	0	1
a_{10}	3	0	3	6	0	0	3	6	3	0	8	6	0	0	3	6
a_{11}	0	3	0	6	0	0	0	6	0	3	0	0	0	0	0	6
a_{12}	0	0	0	0	3	2	0	0	0	2	0	0	5	2	0	0
a_{13}	0	0	0	0	0	0	2	2	3	0	2	2	0	0	2	2
a_{14}	0	0	0	0	4	2	0	0	0	2	0	0	4	2	0	0
a_{15}	0	0	0	0	0	0	0	3	3	0	0	3	0	0	0	6

(b) $382 \cdot p(x, y)$

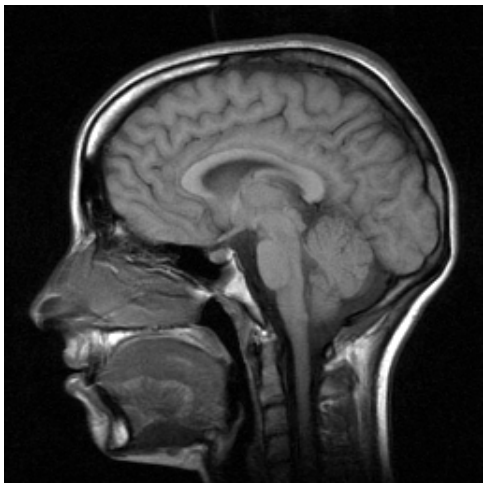
x\ y	a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}
a_0	3	2	0	0	3	2	0	0	3	2	0	0	3	2	0	0
a_1	0	3	0	1	4	3	2	0	0	3	0	1	4	3	2	1
a_2	0	6	0	0	0	6	0	0	0	6	0	0	0	6	0	0
a_3	0	0	3	3	0	0	3	3	0	0	3	3	0	0	3	3
a_4	1	0	2	2	1	1	0	0	0	0	2	2	1	1	0	0
a_5	3	3	0	6	3	3	0	0	3	3	0	6	3	3	0	0
a_6	0	0	6	0	0	0	6	6	0	0	6	0	0	0	6	6
a_7	3	3	0	3	3	3	0	0	3	3	0	3	3	3	0	0
a_8	2	0	2	0	2	0	2	1	2	0	2	0	2	0	2	0
a_9	0	2	0	1	0	2	0	0	0	2	0	1	0	2	0	1
a_{10}	3	0	3	6	0	0	3	6	3	0	3	6	0	0	3	6
a_{11}	0	3	0	6	0	0	0	6	0	3	0	6	0	0	0	6
a_{12}	0	2	0	0	3	2	0	0	0	2	0	1	3	2	0	1
a_{13}	3	0	2	2	0	0	2	2	3	0	2	2	0	0	2	2
a_{14}	0	2	0	0	4	2	0	0	0	2	0	0	4	2	0	0
a_{15}	3	0	0	3	0	0	0	3	3	0	0	3	0	0	0	3

Table 6.3: An example $p(x, y)$ to show that $(\Gamma_1 \cap \Gamma_2) \cup (\Gamma_3 \cap \Gamma_4)$ be traditional UD code is not necessary. Note: * refers to $p(x, y) > 0$.

$\gamma_X(X) \backslash Y$	1	2	3	4
0000	*	0	0	0
0000	0	*	0	0
0000	0	0	*	0
0000	0	0	0	*
001	*	*	0	0
010	*	0	*	0
011	*	0	0	*
011	0	*	*	0
010	0	*	0	*
001	0	0	*	*
100	*	*	*	0
101	*	*	0	*
110	*	0	*	*
111	0	*	*	*
0001	*	*	*	*

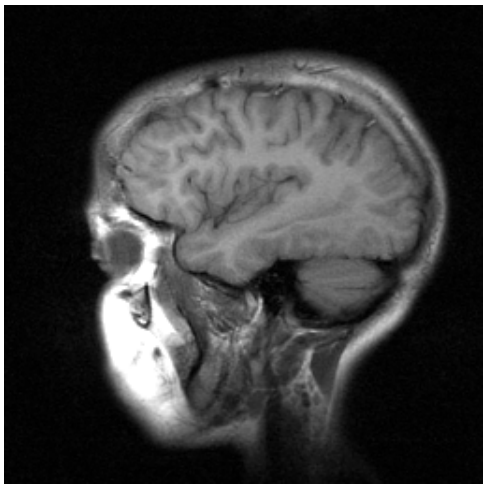


(a) Brain1



(b) Brain2

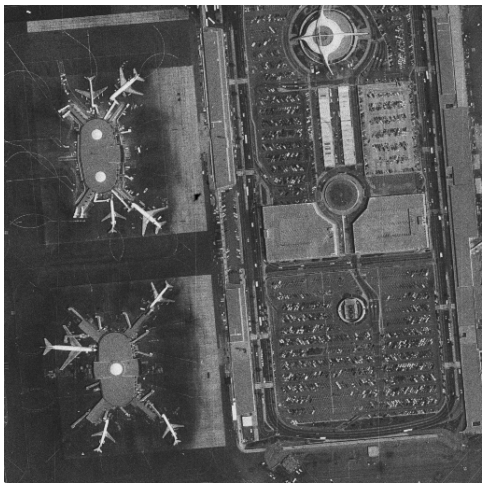
(c) Brain3



(d) Brain4

Figure 6.2: Examples of medical scan images used in ECDQ.

(a) Lena



(b) Airport

(c) Barbara

(d) Crowd

Figure 6.3: Test images used in ECDQ.

Bibliography

[1] A. Kh. Al Jabri and S. Al-Issa. Zero-error codes for correlated information sources. In *Proceedings of Cryptography*, pages 17–22, Cirencester,UK, December 1997.

[2] D. Slepian and J. K. Wolf. A coding theorem for multiple access channels with correlated information sources. *Bell System Technical Journal*, 52:1037–1076, 1973.

[3] R. M. Gray and A. D. Wyner. Source coding for a simple network. *Bell Systems Technical Journal*, 53(9):1681–1721, November 1974.

[4] A. D. Wyner and J. Ziv. The rate-distortion function for source coding with side information at the decoder. *IEEE Trans. Inform. Theory*, 22:1–10, January 1976.

[5] Q. Zhao and M. Effros. Broadcast system source codes: a new paradigm for data compression. In *Conference Record, Thirty-Third Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, CA, October 1999. IEEE. Invited Paper.

[6] Q. Zhao and M. Effros. Lossless and lossy broadcast system source codes: theoretical limits, optimal design, and empirical performance. In *Proceedings of the Data Compression Conference*, pages 63–72, Snowbird, UT, March 2000. IEEE.

[7] M. Fleming, Q. Zhao, and M. Effros. Network vector quantization. *IEEE Transactions on Information Theory*. Revised. Submitted Jan 2002.

[8] Q. Zhao and M. Effros. Optimal code design for lossless and near-lossless source coding in multiple access networks. In *Proceedings of the Data Compression Conference*, pages 263–272, Snowbird, UT, March 2001. IEEE.

[9] Q. Zhao and M. Effros. Lossless source coding for multiple access networks. In *Proceedings of the IEEE International Symposium on Information Theory*, page 285, Washington, DC, June 2001.

[10] Q. Zhao, S. Jaggi, and M. Effros. Side information source coding: low complexity design and source independence. In *Conference Record, 36th Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, CA, November 2002. IEEE. Invited paper.

[11] Q. Zhao and M. Effros. Lossless and near lossless source coding for multiple access networks. *IEEE Transactions on Information Theory*, 49:112–128, January 2003.

[12] Q. Zhao and M. Effros. Low complexity code design for lossless and near lossless side information source codes. In *Proceedings of the Data Compression Conference*, Snowbird, UT, March 2003. IEEE.

[13] Q. Zhao and M. Effros. Necessary and sufficient conditions for uniquely decodable multiple access source codes. In preparation for submission to IEEE Transactions on Information Theory.

[14] Q. Zhao and M. Effros. Necessary and sufficient conditions for uniquely decodable multiple access source codes. In preparation for submission to IEEE International Symposium on Information Theory 2004.

[15] H. Feng, Q. Zhao, and M. Effros. Network source coding using entropy constrained dithered quantization. In *Proceedings of the Data Compression Conference*, Snowbird, UT, March 2003. IEEE.

[16] Q. Zhao, H. Feng, and M. Effros. Multi-resolution source coding using entropy constrained dithered quantization. In preparation for submission to IEEE International Symposium on Information Theory 2004.

[17] Q. Zhao, H. Feng, and M. Effros. Multi-resolution source coding using entropy constrained dithered quantization. In preparation for submission to IEEE Transactions on Information Theory.

[18] M. Fleming and M. Effros. Generalized multiple description vector quantization. In *Proceedings of the Data Compression Conference*, pages 3–12, Snowbird, UT, March 1999. IEEE.

[19] D. Slepian and J. K. Wolf. Noiseless coding of correlated information sources. *IEEE Transactions on Information Theory*, IT-19(4):471–480, July 1973.

[20] Y. Linde, A. Buzo, and R. M. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, 28:84–95, January 1980.

[21] P. A. Chou, T. Lookabaugh, and R. M. Gray. Entropy-constrained vector quantization. *IEEE Transactions on Acoustics Speech and Signal Processing*, 37(1):31–42, January 1989.

[22] H. S. Witsenhausen. The zero-error side information problem and chromatic numbers. *IEEE Transactions on Information Theory*, 22:592–593, 1976.

[23] Y. Yan and T. Berger. On instantaneous codes for zero-error coding of two correlated sources. In *Proceedings of the IEEE International Symposium on Information Theory*, page 344, Sorrento, Italy, June 2000.

[24] N. Alon and A. Orlitsky. Source coding and graph entropies. *IEEE Transactions on Information Theory*, 42(5):1329–1339, September 1996.

[25] S. S. Pradhan and K. Ramchandran. Distributed source coding using syndromes (DISCUS) design and construction. In *Proceedings of the Data Compression Conference*, pages 158–167, Snowbird, UT, March 1999.

[26] S. S. Pradhan and K. Ramchandran. Distributed source coding using syndromes: applications to sensor networks. In *Proceedings of the Data Compression Conference*, Snowbird, UT, March 2000.

[27] T. J. Flynn and R. M. Gray. Encoding of correlated observations. *IEEE Transactions on Information Theory*, IT-33(6):773–787, November 1987.

[28] R. Zamir and S. Shamai. Nested linear/lattice codes for wyner-ziv encoding. In *Information theory workshop*, pages 92–93, Killarney, Ireland, June 1998. IEEE.

[29] J. Bajcsy and P. Mitran. Coding for the Slepian-Wolf problem with turbo codes. In *Proceedings of Globecom*, pages 1400–1404, November 2001.

[30] A. Aaron and B. Girod. Compression with side information using turbo codes. In *Proceedings of the Data Compression Conference*, pages 252–261, Snowbird, UT, March 2002.

[31] Y. Zhao and J. Garcia-Frias. Data compression of correlated non-binary sources using punctured turbo codes. In *Proceedings of the Data Compression Conference*, Snowbird, UT, March 2002.

[32] P. Koulgi, E. Tuncel, S. Regunathan, and K. Rose. Minimum redundancy zero-error source coding with side information. In *Proceedings of the IEEE International Symposium on Information Theory*, page 282, Washington DC, USA, June 2001.

[33] P. Koulgi, E. Tuncel, S. Regunathan, and K. Rose. On zero-error source coding with decoder side information. *IEEE Transactions on Information Theory*, January 2003.

[34] L. Lovász. *Combinatorial problems and exercises, 2nd ed.* North-Holland, Amsterdam, Netherlands, 1993.

[35] S. Even. *Graph Algorithms.* Computer Science Press, Potomac, Maryland, 1979.

[36] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, May 1983.

[37] J. Vaisey and A. Gersho. Simulated annealing and codebook design. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1176–1179, April 1988.

[38] B. McMillan. Two inequalities implied by unique decipherability. *IEEE Transactions on Information Theory*, 2:115–116, 1956.

[39] J. Karush. A simple proof of an inequality of mcmillan. *IRE Trans. Inform. Theory*, 7:118, 1961.

[40] A. A. Sardians and G. W. Patterson. A necessary and sufficient condition for the unique decomposition of coded messages. *IRE Convention Record, Part 8*, pages 104–108, 1956.

[41] S. Even. Tests for unique decipherability. *IEEE Transactions on Information Theory*, 9:109–112, 1963.

[42] S. Even. Tests for synchronizability of finite automata and variable length codes. *IEEE Transactions on Information Theory*, 10:185–189, 1964.

[43] R. M. Capocelli. A necessary and sufficient condition for unique decipherability. *Notices of AMS*, 22:A714, 1975.

[44] S. Jaggi and M. Effros. Universal linked multiple access source codes. In *Proceedings of the IEEE International Symposium on Information Theory*, Lausanne, Switzerland, June 2002.

[45] S. Jaggi and M. Effros. Universal linked multiple access source codes. *IEEE Transactions on Information Theory*. Submitted 2002. In review.

[46] J. Ziv. On universal quantization. *IEEE Trans on Information Theory*, 31(3):344–347, May 1985.

[47] R. Zamir and M. Feder. On universal quantization by randomized uniform/lattice quantizers. *IEEE Transactions on Information Theory*, 38(2):428–436, March 1992.

[48] Y. Frank-Dayan and R. Zamir. Dithered lattice-based quantizers for multiple descriptions. *IEEE Transactions on Information Theory*, 48(1):192–204, January 2002.

[49] H. Feng and M. Effros. Improved bounds for the rate loss of multi-resolution source codes. *IEEE Transactions on Information Theory*, 49(4):809–821, April 2003.

[50] J. Ziv and A. Lempel. A universal algorithm for sequential data compression. *IEEE Trans on Information Theory*, May 1977.

[51] T. Welch. A technique for high-performance data compression. *Computer*, June 1984.

[52] M. Burrows and D.J. Wheeler. A block-sorting lossless data compression algorithm. *Digital Systems Research Center Research Report 124*, 1994.

[53] J. M. Shapiro. Embedded image coding using zerotrees of wavelet coefficients. *IEEE Transactions on Signal Processing*, 41:3445–3462, December 1993.

[54] A. Said and W. A. Pearlman. A new, fast, and efficient image codec based on set partitioning in hierarchical trees. *IEEE Transactions on Circuits and Systems for Video Technology*, 6(3):243–250, June 1996.

[55] S. Y. Tung. *Multiterminal Source Coding*. Ph.D. dissertation, Cornell Univ., Ithaca, NY, 1978.

[56] H. Feng and M. Effros. On the rate loss of multiple description and multiple access source codes. Submitted to the *IEEE Transactions on Information Theory*, May 2002.