

UC San Diego

UC San Diego Electronic Theses and Dissertations

Title

Coding Techniques to Extend the Lifetime of Flash Memories

Permalink

<https://escholarship.org/uc/item/43k8v2hz>

Author

Liu, Yi

Publication Date

2020

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA SAN DIEGO

Coding Techniques to Extend the Lifetime of Flash Memories

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Electrical Engineering
(Communication Theory and Systems)

by

Yi Liu

Committee in charge:

Professor Paul H. Siegel, Chair
Professor Massimo Franceschetti
Professor Young-Han Kim
Professor Ramamohan Paturi
Professor Jason Schweinsberg

2020

Copyright
Yi Liu, 2020
All rights reserved.

The dissertation of Yi Liu is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Chair

University of California San Diego

2020

DEDICATION

To my parents Tong Chang and Sheng Liu

TABLE OF CONTENTS

Signature Page	iii
Dedication	iv
Table of Contents	v
List of Figures	viii
List of Tables	x
Acknowledgements	xi
Vita	xiv
Abstract of the Dissertation	xvi
Chapter 1 Introduction	1
1.1 Flash Memory Architecture	1
1.2 Dissertation Overview	2
Chapter 2 Relevant Works on Shaping Codes and Distribution Matching Codes	5
2.1 Shaping Codes	6
2.1.1 Codes Minimizing Total Cost	6
2.1.2 Rate-Constrained Codes Minimizing Average Cost	9
2.2 Distribution Matching (DM) Codes	11
2.2.1 Applications of DM Codes to Shaping	11
2.2.2 DM Codes with Optimality Measures	12
Chapter 3 Rate-Constrained Shaping Codes for Noiseless Costly Channel	14
3.1 Introduction	14
3.1.1 Shaping Codes	14
3.1.2 Distribution Matching Codes	15
3.2 Information-theoretic Preliminaries	16
3.2.1 Basic Model	16
3.2.2 Asymptotic Symbol Occurrence Probability	18
3.2.3 Lower Bound on Symbol Distribution Entropy	22
3.3 Optimal Shaping Codes	25
3.3.1 Cost Minimizing Probability Distribution	25
3.3.2 Optimal Data Shaping Code Design	32
3.4 Distribution Matching Code Design	44
3.4.1 Problem Formulation	45
3.5 Generalized Expansion Factor	48

3.6	Comparison of DM Performance Measures	52
3.6.1	Generalized Expansion Factor and Informational Divergence	53
3.6.2	Type-I Shaping Problem and Normalized I-Divergence	55
3.7	Experimental Results	61
3.7.1	Optimal Data Shaping Code for MLC Flash Memory	61
3.7.2	Varn Codes for Distribution Matching	65
3.8	Conclusion	66
3.9	Acknowledgment	67
Chapter 4	Direct Shaping Codes for Multi-Level Flash Memories	69
4.1	Shaping Codes for SLC Flash	69
4.1.1	Encoder and Decoder	69
4.1.2	Simulation Results	71
4.2	Data Shaping Codes for MLC Flash	72
4.2.1	Cost Model for MLC Flash	72
4.2.2	Shaping Codes for MLC Flash	74
4.2.3	Encoding Algorithm for MLC Shaping Codes	76
4.2.4	Simulation Results	78
4.2.5	Experimental Results on MLC Flash Memory	79
4.3	Error Propagation Analysis	81
4.3.1	Recurrence Probability of a Two-Word Dictionary	81
4.3.2	Recurrence Probability of a Pair of Words in a General Dic- tionary	85
4.3.3	Recurrence Probability of a General Dictionary	91
4.3.4	Recurrence Behavior of <i>The Count of Monte Cristo</i>	94
4.4	Performance of Direct Shaping Codes	96
4.5	Conclusion	99
4.6	Acknowledgment	100
Chapter 5	Rate-Constrained Shaping Codes for Noiseless Finite-State Costly Channels	101
5.1	Capacity of the Noiseless Finite-State Costly Channel	101
5.1.1	Channel Capacity with Average Cost Constraint	102
5.1.2	Channel Capacity without Cost Constraint	104
5.2	Finite-State Variable-Length Codes: A Word-Valued Source Approach	105
5.2.1	Finite-State Variable-Length Codes	105
5.2.2	Asymptotic Performance	109
5.2.3	Finite-State Word-Valued Source	110
5.2.4	Asymptotic Normalized KL-Divergence	115
5.3	Optimal Shaping Codes for Finite-State Costly Channel	118
5.4	Optimal Shaping Code Design	132
5.4.1	Equivalence Theorem	132
5.4.2	Generalized Varn Code	137
5.4.3	Separation Theorem	141

5.5	Markov Distribution Matching Code Design	142
5.5.1	Problem Formulation	142
5.6	Generalized Expansion Factor	145
5.6.1	Motivation and Definition	145
5.7	Acknowledgment	148
Chapter 6	Bad Page Detector for NAND Flash Memories	149
6.1	Introduction	149
6.2	Bad Page Detector	149
6.2.1	P/E Cycling Experiment	150
6.2.2	(T, T_{Of}, N_{Th}) –Bad Page	150
6.2.3	Bad Page Detector Overview	154
6.3	Time-Dependent Neural Network	154
6.3.1	Network Structure	155
6.3.2	Approximating the Time Dependent Dense Layer	156
6.3.3	Training the Network	157
6.4	Long-Short Term Memory	158
6.5	Experiment Results	158
6.5.1	Q -Bad Page	158
6.5.2	Analysis of Results	159
6.6	Acknowledgment	162
Bibliography		163

LIST OF FIGURES

Figure 3.1:	Total cost versus f for random source with $\mathcal{C} = [1, 2, 3, 4]$	28
Figure 3.2:	Achievable total cost vs. expansion factor	41
Figure 3.3:	Probability of occurrence $\hat{P}_0$ of a Varn code for the target distribution $\{2/3, 1/3\}$	50
Figure 3.4:	Generalized expansion factor of a Varn code for the target distribution $\{2/3, 1/3\}$	51
Figure 3.5:	Codeword length distribution of Varn code with the codebook size $K = 256$ for English-language text.	62
Figure 3.6:	BER performance for English-language text.	63
Figure 3.7:	Codeword length distribution of Varn code with codebook size $K = 256$ for Chinese-language text.	64
Figure 3.8:	BER performance for Chinese-language text.	65
Figure 4.1:	Direct shaping codes applied to <i>The Count of Monte Cristo</i>	70
Figure 4.2:	Schematic of MLC flash cell programming.	73
Figure 4.3:	Cell error rate of MLC Flash for different programming levels, while repeatedly programming a constant level.	73
Figure 4.4:	Fractions of MLC cell levels for segments of <i>The Count of Monte Cristo</i> without a shaping code.	78
Figure 4.5:	Fractions of MLC cell levels for segments of <i>The Count of Monte Cristo</i> with MLC shaping code.	79
Figure 4.6:	BER performance of English-language text <i>The Count of Monte Cristo</i> . . .	80
Figure 4.7:	BER performance of Chinese-language text <i>Collected Works of Lu Xun, Volumes 1–4</i>	80
Figure 4.8:	Error propagation examples.	81
Figure 4.9:	Two dimensional random walk on (N_i^e, N_i^d) -plane	84
Figure 4.10:	Two dimensional random walk in a general dictionary	86
Figure 4.11:	Recurrence probabilities of a two-word dictionary with distribution $P = \{0.6, 0.4\}$ and transition probability $\rho = 0.05$	90
Figure 4.12:	Upper bound and simulation result of $P(t)$ with different transition probability ρ	94
Figure 4.13:	Fraction of words for segments of <i>The Count of Monte Cristo</i>	95
Figure 4.14:	Probability of decoding recurrence when segments of <i>The Count of Monte Cristo</i> are decoded correctly.	95
Figure 5.1:	Flash memory channel	102
Figure 5.2:	Encoding process.	106
Figure 5.3:	Achievable total cost vs. expansion factor.	135
Figure 5.4:	The total cost of the subcodebooks of a generalized Varn code on the modified channel with different starting state.	140

Figure 5.5:	The total cost of a generalized Varn code on the original SLC flash memory channel	140
Figure 6.1:	Measured bit error counts vs. P/E cycle count of lower pages.	151
Figure 6.2:	Measured bit error counts vs. P/E cycle count of middle pages.	151
Figure 6.3:	Measured bit error counts vs. P/E cycle count of upper pages.	152
Figure 6.4:	Example of an (T, T_{Of}, N_{Th}) -bad page.	153
Figure 6.5:	Bad Page Detection System.	154
Figure 6.6:	The structure of TDNN.	156
Figure 6.7:	The realization of time dependent dense layer. DLs represent dense layers.	157
Figure 6.8:	Distribution of Q -bad pages under different input vector-length and network structures.	160
Figure 6.9:	Functions in the cubic TDNN with input length 5.	161
Figure 6.10:	Error analysis	162

LIST OF TABLES

Table 3.1:	First-, second-, and third-order KL-divergence	66
Table 4.1:	Length-2 dictionary when encoding $\mathbf{x} = 00$	71
Table 4.2:	Ordered list of upper page words for lower page codeword 1110.	76
Table 4.3:	Corresponding list of cell level words for lower page codeword 1110.	76
Table 4.4:	Encoding and decoding processes in Fig. 4.8a.	82
Table 4.5:	Encoding and decoding processes in Fig. 4.8b.	83
Table 4.6:	Transition probability of a random walk in a general dictionary.	87
Table 5.1:	Flash memory channel Cost	102
Table 5.2:	Optimal symbol occurrence probabilities for flash memory channel.	131
Table 5.3:	Cost table of DNA synthesis.	131
Table 5.4:	Modified cost for flash memory channel.	141
Table 6.1:	Performance of TDNN, LSTM and SVM.	161

ACKNOWLEDGEMENTS

This dissertation would not have been possible without the help from many people.

First, I would like to express my gratitude and appreciation to my advisor, Prof. Paul H. Siegel, for his guidance, encouragement, and patience throughout my Ph.D. studies. It was his fascinating courses that introduced me to the field of coding theory. I was inspired by his vast knowledge and experience, and his attention to details taught me the correct way of thinking. More importantly, he gave me enough freedom to tackle problems in various aspects. Sharing new idea with him is always full of fun. I would also like to thank him for inviting me to join his family for thanksgiving dinners. His kindness and generosity taught me how to become a better person.

I would like to thank Prof. Laurence Milstein, Prof. Alon Orlitsky, Prof. Massimo Franceschetti and Prof. Jason Schweinsberg for their courses on communication theory, information theory, and probability theory. This dissertation benefits a lot from the knowledge I have gained from these courses. I am also thankful to Prof. Paul H. Siegel, Prof. Massimo Franceschetti, Prof. Young-Han Kim, Prof. Ramamohan Paturi, and Prof. Jason Schweinsberg for their time and effort in serving on my committee.

It is my pleasure, and an honour, to participate in multiple collaborative projects. I would like thank my collaborators: Pengfei Huang, Yonglong Li, Eitan Yaakobi, Andreas Lenz, Alexander W. Bergman, Erich F. Haratsch, Cyrus Rashtchian, and Si Wu. It has been a great pleasure working with them.

I would like to thank Xiaojie Zhang, my mentor during the internship at CNEX Labs, and Hironori Uchikawa, my mentor during the internship at Kioxia. I also want to thank Naoaki Kokubun for his help during my internship.

I would like to thank the current and former STAR group members: Pengfei Huang, Xiaojie Zhang, Eitan Yaakobi, Veeresh Taranalli, Bing Fan, Yonglong Li, Wei Wu, Karthik Nagarjuna Tunuguntula, Shahar Dror, Hironori Uchikawa, Osamu Torii, Naoaki Kokubun, Lingjun Kong,

Sarit Buzaglo, Simeng Zheng, Navya Sree Prem, Yang Qiu, Yishen Yeh, Alexander W. Bergman, Zachary Blair, Si Wu, Iris Lau, Michael Baluja, and Zhen Zhai. They created an intellectual and friendly environment to collaborate and conduct research. I would like to give a special thanks to Pengfei Huang, who shared the same office room with me for more than three years. His guidance on many parts of this dissertation is important. I would also like to thank CMRR staff: Iris Villanueva, Marina Robenko, Ray Descoteaux, and Octavio Ochoa, for their support during my Ph.D. studies.

I would like to thank my roommates: Jun Wang, Keming Cao, and Bentao Zhang, for their company when life was hard. I also want to thank my friends at UCSD and OriginalFire. Without their support, this dissertation would not have been possible.

This dissertation is dedicated to my parents, Ms. Tong Chang and Mr. Sheng Liu, who give me life, wisdom, and unconditional love.

This dissertation was supported in part by National Science Foundation (NSF) Grant CCF-1619053, the Center for Memory and Recording Research at UC San Diego, and Toshiba Corporation.

Chapter 3 is in part a reprint of the material in the paper: Yi Liu, Pengfei Huang, Alexander W. Bergman, and Paul H. Siegel, Rate-Constrained Shaping Codes for Structured Sources, *IEEE Transactions on Information Theory*, pp. 1-23, Apr. 2020 (Early Access). It is also in part a reprint of materials in the papers: Yi Liu, Pengfei Huang, and Paul H. Siegel, “Performance of Optimal Data Shaping Codes,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Aachen, Germany, Jun. 25-30, 2017, pp. 1003–1007, Yi Liu, Pengfei Huang, Alexander W. Bergman, and Paul H. Siegel, “Optimal Data Shaping Code Design,” in *Proc. 9th Annu. Non-Volatile Memories Workshop (NVMW)*, La Jolla, CA, Mar. 11-13, 2018, and Yi Liu, Pengfei Huang, and Paul H. Siegel, “Performance of Shaping Codes for Flash Memory,” in *Proc. 8th Annu. Non-Volatile Memories Workshop (NVMW)*, La Jolla, CA, Mar. 12–14, 2017. The dissertation author was the primary investigator and author of these papers.

Chapter 4 is in part a reprint of the material in the paper: Yi Liu and Paul H. Siegel, On the Performance of Direct Shaping Codes, which has been submitted to *IEEE Transactions on Communications*. It is also in part a reprint of materials in the papers: Yi Liu and Paul H. Siegel, “Shaping Codes for Structured Data,” in *Proc. IEEE Globecom*, Washington, D.C., Dec. 4-8, 2016, pp. 1–5, and Yi Liu and Paul H. Siegel, “Shaping Codes for Structured Data,” in *Proc. 7th Annu. Non-Volatile Memories Workshop (NVMW)*, La Jolla, CA, Mar. 6-8, 2016. The dissertation author was the primary investigator and author of these papers.

Chapter 5 is in part a reprint of the material in the paper: Yi Liu, Yonglong Li, Pengfei Huang, and Paul H. Siegel, Rate-Constrained Shaping Codes for Noiseless Finite-State Costly Channels, which is being prepared for submission to *IEEE Transactions on Information Theory*. The dissertation author was the primary investigator and author of this paper.

Chapter 6 is in part a reprint of the material in the paper: Yi Liu, Si Wu, and Paul H. Siegel, Bad Page Detector for NAND Flash Memory, in *Proc. 11th Annual Non-Volatile Memories Workshop (NVMW)*, La Jolla, CA, USA, Mar. 2020. The dissertation author was the primary investigator and author of this paper.

VITA

2014	Bachelor of Science in Physics, Peking University, China
2016	Master of Science in Electrical Engineering (Communication Theory and Systems), University of California San Diego
2020	Doctor of Philosophy in Electrical Engineering (Communication Theory and Systems), University of California San Diego

PUBLICATIONS

Yi Liu, Pengfei Huang, Alexander W. Bergman, and Paul H. Siegel, “Rate-Constrained Shaping Codes for Structured Sources”, *IEEE Trans. Inf. Theory*, pp. 1-23, Apr. 2020 (Early Access).

Yi Liu and Paul H. Siegel, “On the Performance of Direct Shaping Codes”, submitted to *IEEE Trans. Commun.*

Yi Liu, Yonglong Li, Pengfei Huang, and Paul H. Siegel, “Rate-Constrained Shaping Codes for Noiseless Finite-State Costly Channels,” to be submitted to *IEEE Trans. Inf. Theory*.

Pengfei Huang, Yi Liu, Xiaojie Zhang, Paul H. Siegel, and Erich F. Haratsch, “Syndrome-Coupled Rate-Compatible Error-Correcting Codes: Theory and Application,” *IEEE Trans. Inf. Theory*, vol. 66, no. 4, pp. 2311-2330, Apr. 2020.

Andreas Lenz, Yi Liu, Cyrus Rashtchian, Paul H. Siegel, Antonia Wachter-Zeh, and Eitan Yaakobi, “Coding for Efficient DNA Synthesis,” to appear in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Los Angeles, California, USA, Jun. 2020.

Pengfei Huang, Yi Liu, Xiaojie Zhang, Paul H. Siegel, and Erich F. Haratsch, “Syndrome-coupled rate-compatible error-correcting codes,” in *Proc. IEEE Inf. Theory Workshop (ITW)*, Kaohsiung, Taiwan, Nov. 2017, pp. 454-458.

Yi Liu, Pengfei Huang, and Paul H. Siegel, “Performance of Optimal Data Shaping Codes,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Aachen, Germany, Jun. 25-30, 2017, pp. 1003–1007.

Yi Liu and Paul H. Siegel, “Shaping Codes for Structured Data,” in *Proc. IEEE Globecom*, Washington, D.C., Dec. 4-8, 2016, pp. 1–5.

Yi Liu, Si Wu, and Paul H. Siegel, “Bad Page Detector for NAND Flash Memory,” in *Proc. 11th Annu. Non-Volatile Memories Workshop (NVMW)*, La Jolla, CA, Mar. 8-10, 2020.

Yi Liu, Pengfei Huang, Alexander W. Bergman, and Paul H. Siegel, “Optimal Data Shaping Code Design,” in *Proc. 9th Annu. Non-Volatile Memories Workshop (NVMW)*, La Jolla, CA, Mar. 11-13, 2018.

Yi Liu, Pengfei Huang, and Paul H. Siegel, “Performance of Shaping Codes for Flash Memory,” in *Proc. 8th Annu. Non-Volatile Memories Workshop (NVMW)*, La Jolla, CA, Mar. 12–14, 2017.

Yi Liu and Paul H. Siegel, “Shaping Codes for Structured Data,” in *Proc. 7th Annu. Non-Volatile Memories Workshop (NVMW)*, La Jolla, CA, Mar. 6-8, 2016.

ABSTRACT OF THE DISSERTATION

Coding Techniques to Extend the Lifetime of Flash Memories

by

Yi Liu

Doctor of Philosophy in Electrical Engineering
(Communication Theory and Systems)

University of California San Diego, 2020

Professor Paul H. Siegel, Chair

NAND flash memory has become a widely used data storage technology. It uses rectangular arrays, or *blocks* of floating-gate transistors (commonly referred to as *cells*) to store information. The flash memory cells gradually wear out with repeated programing and erasing, referred to as *program/erase (P/E) cycling*, but the damage caused by P/E cycling is dependent on the programmed cell level [43], [44]. For example, in SLC flash memory, each cell has two different states, erased and programmed, represented by 1 and 0, respectively. Storing 1 in a cell causes less damage, or *wear*, than storing 0. More generally, in multilevel flash memories, the cell wear is an increasing function of the programmed cell level. The main research goal of this

dissertation is to design new coding techniques that can extend the lifetime of flash memories.

The damage caused by programming the cell is usually modeled as a *cost*, and increasing the lifetime of flash memories can be converted to the problem of encoding information for use on channels with a cost constraint. This type of code is often referred to as a *shaping code*. Therefore in this dissertation we study rate-constrained shaping codes for noiseless costly channels. We systematically investigate the fundamental performance limits of fixed-to-variable length shaping codes from a rate and distribution perspective for a memoryless channel. Then, we study a recently proposed rate-1 *direct* shaping code and study its error propagation property. In addition, we consider shaping codes for finite-state channels.

One observation from the above analysis is that an optimal shaping code for a memoryless channel generates a codeword sequence that approximates an i.i.d. process, and an optimal shaping code for a finite-state channel generates a sequence that approximates a stationary Markov process. In this dissertation, we study the connection between shaping codes and distribution matching codes that map a sequence of i.i.d. source symbols into an output sequence that approximates an i.i.d. or a stationary Markov process.

In the flash memory device, the *bit error count (BEC)* behavior varies significantly among pages. Therefore we propose a *bad page detector*, which predicts whether a page will become a “bad” page in the near future based on its current and previous BEC information. Two machine learning algorithms, based upon time-dependent neural network and long-short term memory architectures, are used to design the detector.

Chapter 1

Introduction

1.1 Flash Memory Architecture

NAND flash memory has become a widely used data storage technology. It uses rectangular arrays, or *blocks* of floating-gate transistors (commonly referred to as *cells*) to store information. The basic storage unit in a NAND flash memory is a floating-gate transistor, commonly referred to as a *cell*. The voltage levels of a cell can be adjusted by a program operation and are used to represent the stored data. The cells can typically be programmed to 2, 4, 8, and 16 voltage levels (1, 2, 3, and 4 bits/cell, respectively) and are referred to as single-level cell (SLC), multi-level cell (MLC), triple-level cell (TLC), and quad-level cell (QLC). Cells are organized into a rectangular array, interconnected by horizontal wordlines and vertical bitlines. A flash memory chip comprises a collection of such blocks. Blocks are the basic units for an erase operation.

In the TLC flash memory, the three bits in the cells connected along a wordline are assigned to three separate *pages* (lower, middle, and upper pages), which represent the basic unit for program and read operations. The cells gradually wear out with repeated programming and erasing, referred to as *program/erase (P/E) cycling*, resulting in an increase in a page's bit

error counts (BECs). The damage caused by P/E cycling is dependent on the programmed cell level [43], [44], and the BEC performance of pages within a block varies significantly. In this dissertation we propose two methods that can combat the limited-endurance in flash memory. The first method models the damage caused by programming the cell as a cost. Therefore extending the lifetime of flash memories can be converted to the problem of encoding information for use on channels with a cost constraint. This type of code is often referred to as a *shaping code*, and can trace its origins back to Shannon [66]. The second method improves the lifetime of a flash device by identifying and retiring the pages that wear out and storing data in the remaining part of the device. The algorithm that predicts a page’s BEC behavior is referred to as a *bad page detector*.

1.2 Dissertation Overview

In this dissertation, we first study shaping codes for noiseless costly channels. Then, we propose two bad page detectors that are used to extend the lifetime of flash memory devices. The dissertation is organized as follows

In Chapter 2, we provide a brief review of relevant work on shaping codes and distribution matching codes.

In Chapter 3, we study rate-constrained fixed-to-variable length shaping codes for noiseless, memoryless costly channels and arbitrary i.i.d. sources. The analysis relies on the theory of word-valued sources. We establish a relationship between the code expansion factor – the ratio of the expected codeword length to the length of the input source word – and the minimum average symbol cost. We then determine the expansion factor that minimizes the average cost per source symbol (total cost), corresponding to a conventional optimal source code with cost. An equivalence is established between codes minimizing average symbol cost and codes minimizing total cost, and a separation theorem is proved, showing that optimal shaping can be achieved by a concatenation of optimal compression and optimal shaping for a uniform i.i.d. source.

Shaping codes often incorporate, either explicitly or implicitly, some form of non-equiprobable signaling. We use our results to further explore the connections between shaping codes and codes that map a sequence of i.i.d. source symbols into an output sequence of symbols that are approximately independent and distributed according to a specified target distribution, such as distribution matching (DM) codes. Optimal DM codes are characterized in terms of a new performance measure - generalized expansion factor (GEF) - motivated by the costly channel perspective. The GEF is used to study DM codes that minimize informational divergence and normalized informational divergence.

In Chapter 4, we study direct shaping codes for SLC flash memory. We describe an adaptation of this coding algorithm that provides data shaping for MLC flash memory. It makes use of a page-dependent cost model and is designed to be compatible with the standard procedure of row-by-row, page-based, wordline programming. We also give experimental results demonstrating the performance of MLC data shaping codes when applied to English and Chinese language texts. We then study the potential error propagation properties of direct shaping codes when used in a noisy flash device. In particular, we model the error propagation as a biased random walk in a multidimensional space. We prove an upper bound on the error propagation probability and propose an algorithm that can numerically approach a lower bound. Finally, we study the asymptotic performance of direct shaping codes. We prove that the SLC direct shaping code is suboptimal in the sense that it can only achieve the minimum average cost for a rate-1 code under certain conditions on the source distribution.

In Chapter 5, we study rate-constrained shaping codes for finite-state noiseless costly channels and arbitrary i.i.d. sources. We establish a relationship between the code expansion factor and minimum average symbol cost. We then determine the expansion factor that minimizes the average cost per source symbol (total cost), corresponding to a conventional “optimal source code with cost”. An equivalence is established between codes minimizing average symbol cost and codes minimizing total cost, and a separation theorem is proved, showing that optimal shaping

can be achieved by a concatenation of optimal compression and optimal shaping for a uniform i.i.d. source. Based on the separation theorem, we propose generalized Varn codes that can be used to construct asymptotically optimal shaping codes. We then use our results to further explore the connections between shaping codes and Markov distribution matching codes, which map a sequence of i.i.d. source symbols into an output sequence of symbols that approximates a finite-order Markov process with specified target transition probabilities.

In Chapter 6, we design the bad page detector, which predicts whether a page will become a “bad” page in the near future based on its current and previous BEC information. Several machine learning algorithms, including time-dependent neural network and long-short term memory, are used to design the detector. The experimental results based on data generated from our TLC flash test platform are also presented.

Chapter 2

Relevant Works on Shaping Codes and Distribution Matching Codes

Shaping codes are used to encode information for use on channels with a cost constraint. A prominent application is in data transmission with a power constraint, where constellation shaping is achieved by addressing into a suitably designed multidimensional constellation or, equivalently, by incorporating, either explicitly or implicitly, some form of non-equiprobable signaling. An excellent reference on this topic is Fischer [16].

More recently, shaping codes have been proposed for use in data storage on flash memories subject to a constraint on memory cell wear. In that application, storage system requirements often impose a rate constraint, and the data source may be structured, rather than unconstrained. Motivated by this scenario, this paper investigates information-theoretic properties and design of rate-constrained fixed-to-variable length shaping codes for noiseless, memoryless costly channels and general i.i.d. sources. The analysis relies on the theory of word-valued sources developed in Nishiara and Morita [56]. Our primary interest is in the design of codes that minimize the average cost per code symbol for a given expansion factor – the ratio of the expected codeword length to the length of the input source word – which we refer to as the *type-I shaping problem*.

We also consider the well-studied problem of designing codes that minimize average cost per code symbol, or *total cost*, which we refer to as the *type-II shaping problem*.

The word-valued source analysis provides a natural link between shaping codes and codes that efficiently map a sequence of i.i.d. source symbols into an output sequence of symbols that are approximately independent and distributed according to a specified target distribution. Such codes have been studied in the context of random number generating source codes by Han and Uchida [26] and as distribution matching (DM) codes by Böcherer and Mathar [8], Böcherer [5], Amjad and Böcherer [3], Böcherer and Amjad [6], Schulte and Böcherer [64] and Schulte and Steiner [65]. Our shaping code analysis suggests a new performance measure - generalized expansion factor (GEF) - for fixed-to-variable length DM codes which we use to study codes that minimize informational divergence and normalized informational divergence from a shaping code perspective.

There is a substantial literature on shaping codes and, more recently, a body of work relating to DM codes. Therefore, before summarizing our results in more detail, we provide a brief review of relevant work in both of these areas as a framework for our contributions.

2.1 Shaping Codes

2.1.1 Codes Minimizing Total Cost

The problem of coding for noiseless costly channels, or source coding with unequal symbol costs, traces its conceptual origins to Shannon’s 1948 paper that launched the study of information theory [66]. In that paper, Shannon considered the problem of transmitting information over a telegraph channel. The channel symbols – dots and dashes – have different time durations, which can be interpreted as transmission costs. Shannon determined the symbol probabilities that maximize the data transmission rate with integer symbol costs. This result was then generalized to arbitrary positive symbol costs by Krause [38] and Csiszár [13]. In [35], a new

way to calculate the channel capacity was introduced. The ratio of entropy rate and average edge cost of a stationary Markov process is defined as the probabilistic capacity of the channel, and the symbol probabilities that maximize the probabilistic capacity was determined. The authors then proved that the probabilistic capacity equals to the combinatorial capacity. In [5] and [9], the graph that is dense was studied and the equivalence between the combinatorial and probabilistic capacities was also established.

Several researchers have considered the problem of designing codes for memoryless costly channels with an i.i.d. source. Most of this work has emphasized construction of codes that minimize average cost per source symbol, which we refer to as *total cost*, without an explicit rate constraint. In Karp [34], costly channel coding was studied from an algebraic perspective, and the problem of designing a shaping code to minimize total cost was recast as an integer programming problem. However, this code design approach is not computationally practical, and the algorithm proposed to reduce the complexity will result in sub-optimal results. In Golin et al. [21], a dynamic programming solution for this integer programming problem was proposed, providing a polynomial time bound on complexity. Other approaches using tree-based constructions were proposed in [38], Melhorn [54], and Csiszár and Körner [14]. They all constructed asymptotically optimal prefix-free variable-length shaping codes. A universal coding scheme based on types was also introduced in [14].

A special case, corresponding to a uniform i.i.d. source in which all codewords are equally likely to occur, was studied by Varn [78], who proposed a variable-length code construction that minimizes the average codeword cost for a fixed codebook size. This coding technique was then incorporated into a universal coding scheme in Iwata [30], which combines LZ78 compression with Varn coding. Later in this paper, we generalize the Iwata scheme, which can be viewed as an embodiment of a separation theorem proved in Section 3.3, and further explore properties and applications of Varn codes.

A generalization of Huffman coding for unequal symbol costs was proposed in Gilbert

[20]. In Guazzo [23], practical arithmetic coding was introduced. This coding technique was then generalized by Savari and Gallager [62] and its properties, such as optimality and coding delay, were analyzed. However, the analysis is based on infinite precision arithmetic coding, which cannot be realized in practice.

In Böcherer and Mathar [8] and Böcherer [5], a variable-to-fixed length code construction called geometric Huffman coding was used to design codes for an i.i.d. uniform source that asymptotically minimize the total cost of a noiseless channel with unequal symbol durations. (This construction matches codeword probabilities to dyadic symbol distributions that optimally approximate the optimal symbol distribution.)

Many works have considered the problem of designing codes for finite-state noiseless costly channel. In [13], the modified Shannon-Fano code based on matching the probability of source and codeword sequences was introduced. The author also proved that this type of codes are asymptotically optimal. The same idea was used in [62], where an arithmetic coding technique was introduced.

Several works extended shaping codes that are designed for memoryless channels to finite-state channels. In [5], the author transformed a finite-state graph to its memoryless representation $\mathcal{M}$ and designed an asymptotically capacity-achieving code on $\mathcal{M}$ using a normalized geometric Huffman code. In [30], the author extended the dynamic programming algorithm introduced in [21] to finite-state channels. The proposed algorithm constructs a local optimal code for each starting state. Further, in [18], an iterative algorithm that can construct globally optimal codes was proposed.

We emphasize that all of the codes mentioned above considered the problem of minimizing cost per source symbol, i.e., total cost, with no explicit consideration of rate. The dependence of total cost on code rate was not thoroughly investigated.

2.1.2 Rate-Constrained Codes Minimizing Average Cost

The problem of designing rate-constrained codes for costly channels has received less attention. For a finite-state channel with associated symbol/transition costs and an average cost constraint, the maximum entropy of a stationary Markov chain, along with the entropy-maximizing symbol/transition probabilities, can be found in McEliece and Rodemich [52], Justesen and Høholdt [32], and Khandekar, McEliece, and Rodemich [35]. In McEliece [51], the special case corresponding to a memoryless channel is addressed. Later in Böcherer [5], both memoryless channels and generalizations to channels with memory were addressed. Getting the entropy-maximizing distribution of the stationary Markov process is a convex optimization problem. The result depends on the *one-step cost-enumerator* matrix $D(B) = \{d_{ij}(B)\}$, where $d_{ij}(B) = 2^{-Bu_{ij}} u_{ij}$ is the cost of edge from state i to j . The capacity depends on the analysis of strictly log-convexity of the Perron root [49] of $D(B)$, which is not well studied in those papers.

Böcherer and Mathar [8] and Böcherer [5] apply the geometric Huffman coding approach to design variable-to-fixed length codes that match codeword probabilities to dyadic symbol distributions that approximate the entropy-maximizing probability mass function for memoryless costly channels subject to an average cost constraint, thereby asymptotically achieving the maximum rate.

The state-splitting algorithm [2], which was developed to construct finite-state codes for constrained channels, has been extended for application to construction of codes for costly channels. Heegard, Marcus, and Siegel [27] studied a class of channels with average runlength constraints, which represent a special case of noiseless channels with a cost constraint. They constructed variable-to-variable length synchronous codes using state-splitting techniques adapted for channels with variable-length symbols. Khayrallah and Neuhoff [36] and McLaughlin and Khayrallah [53] construct fixed-to-fixed length and variable-to-fixed length codes based on state-splitting methods for magnetic recording and constellation shaping applications. Krachkovsky et al. [37] determine a costly channel model matched to a Markov source and construct corre-

sponding codes using enumerative techniques for application to transmission over an intersymbol-interference channel. All of these works strive to construct codes that come close to the capacity-cost functions originally presented in [51],[52], and [32]. Other constructions can be found in [37], [68] and [69].

Other recent work relating to this problem has been motivated by non-volatile memory applications, so we briefly describe the corresponding costly channel model. NAND flash memory uses floating-gate transistors, commonly referred to as *cells*, to store information in the form of different cell voltage levels. The flash memory cells gradually wear out with repeated writing and erasing, referred to as program/erase cycling, and the damage caused by the cycling is dependent on the programmed voltage levels [43], [44]. The costly channel model associates to each cell voltage level a wear cost reflecting the extent of the damage induced by writing that level.

Recently, in [31], Jagmohan et al. proposed *endurance coding*, intended for shaping of programmed data for flash memories. For a given cost model and a specified target code rate, the optimal distribution of cell levels that minimizes the average cost was determined analytically, reproducing the results in the references cited above. For single bit per cell (SLC) flash memory, with associated level costs of 0 and 1, greedy enumerative codes that minimize the number of cells with cost 1 were designed and evaluated in terms of the rate-cost trade-off. However, endurance coding is intended for uniform i.i.d. source data. For structured source data, which would include a general i.i.d. source, the idea of combining source compression with endurance coding was proposed, but the relationship between the code performance and the code rate for arbitrary sources was not thoroughly studied.

In Sharon et al. [67], low-complexity, rate-1, fixed-length *direct shaping codes* for structured data were proposed for use on SLC flash memory. The code construction used a greedy approach based upon an adaptively-built encoding dictionary that does not require knowledge of the source statistics. This construction was extended to a direct shaping code compatible with two-bit per cell (MLC) flash memory operation by Liu et al. in [43], [44]. However, it was proved

in Liu and Siegel [46] that direct shaping codes are in general suboptimal. (Our experimental results in Section 3.7 contain a comparison of a shaping scheme motivated by our analysis to a direct shaping code on MLC flash memory.)

2.2 Distribution Matching (DM) Codes

2.2.1 Applications of DM Codes to Shaping

The application of non-equiprobable signaling in the context of coding with a cost constraint reflects the interesting interplay between shaping codes and DM-type codes (in the broad sense of codes that map an i.i.d. sequence of source symbols to an output sequence of symbols that are approximately independent and distributed according to $\{P_i\}$). Beginning with the work on constellation shaping, there have been a number of applications of DM-type codes to coding for a costly channel.

In [17], signal constellations with non-uniform symbol probabilities were used for efficient modulation on band-limited channels. “Noting the conceptually dual nature of non-equiprobable signaling and source coding, as articulated in [16, Chapter 4], several authors proposed the use of “reverse” source codes derived from, for example, Huffman codes, Tunstall codes, and arithmetic codes as approximate DM codes for shaping applications. See, for example, works by Kschischang and Pasupathy [39], Ungerboeck [77], and Abrahams [1]. Limitations in trying to establish a precise duality between source codes and DM codes are discussed in [8], [3], Baur and Böcherer[4], Lempel et al. [40], and [5, Section 3].”

Gallager [19, p. 208] proposed a method of generating symbols with a biased distribution to be combined with linear coding as an approach to achieving capacity of an asymmetric channel. This idea was incorporated into a general scheme that can use capacity-achieving codes for symmetric channels, such as polar codes, to achieve the capacity of arbitrary discrete memoryless asymmetric channels in Mondelli et al. [55].

In [10], Böcherer et al. propose a scheme that combines DM codes (such as constant composition codes) with systematic error correction codes. This scheme can be regarded as a simplification of the bootstrap scheme in Böcherer and Mathar [7], which concatenates the check bits generated by the systematic ECC encoder with the following information bits and applies a DM encoder to them. In [55], the authors also proved that the bootstrap scheme, which they refer to as a chaining construction, can be used to achieve the capacity of any discrete memoryless asymmetric channel.

2.2.2 DM Codes with Optimality Measures

In Han [25] and Visweswariah et al. [79], it was shown that an optimal variable-length source code can be regarded as an optimal variable-length DM code for a uniform distribution. The criterion for optimality was the vanishing of a form of normalized conditional Kullback-Leibler (KL) divergence between a subset of codewords of fixed length and words generated i.i.d. with the target distribution, asymptotically in the block length. This result was further developed in Han and Uchida [26], where an optimal variable-length source code with cost, meaning a code that minimizes total cost, was shown to be an optimal DM code. The maximum achievable rate of non-prefix-free DM codes was discussed in Uchida [76].

In [8], dyadic probability mass functions with some optimality properties were used to match the capacity-achieving probability distribution of a discrete memoryless channel, and variable-to-fixed length geometric Huffman codes, mentioned earlier, were used as DM codes. Normalized informational divergence – defined as the KL-divergence between a codeword probability distribution and the distribution of the codewords when generated i.i.d. by the target distribution, normalized by the codeword length – was introduced as the DM code optimality measure. It was then proved that geometric Huffman coding is asymptotically optimal, in the sense that the normalized informational divergence converges to zero as the codeword length increases. Other fixed-length DM codes with vanishing normalized informational divergence

were presented in Ramabadran [58] and [64].

Constellation shaping techniques have also been adapted for use in DM coding. For example, a divergence-optimal DM code based on shell mapping was presented in [65] and a DM-like code using trellis-based enumerative amplitude shaping was presented by Gültekin et al. [24].

In [3], the notions of informational divergence and normalized information divergence were extended to measure the performance of fixed-to-variable length codes. Optimality of complete Tunstall code trees with respect to minimizing informational divergence was proved, a result we extend in Section 3.6. An efficient algorithm for finding binary DM codes that minimize the normalized informational divergence, based on an iterative adaptation of binary Tunstall coding, was presented, and asymptotic optimality with increasing block length was established. In [6] the relationship between normalized information divergence of a DM code and its rate was studied, a topic that we further address in Section 3.6.

Chapter 3

Rate-Constrained Shaping Codes for Noiseless Costly Channel

3.1 Introduction

3.1.1 Shaping Codes

In this Chapter, our goal is to systematically study the fundamental performance limits of fixed-to-variable length shaping codes from a rate and distribution perspective. We first use known properties of word-valued sources to determine the symbol occurrence probability of shaping code output sequences (Lemma 4). We then derive an upper bound on the code sequence entropy rate (Lemma 5). Using these results, we are able to reduce the problem of minimizing average code symbol cost subject to a constraint on the code rate to an optimization problem for an i.i.d. process. This problem can be viewed as the dual problem to the entropy-maximization problem considered in the prior literature. We refer to this minimization problem as the *type-I shaping problem*, and we call shaping codes that achieve the minimum average cost for a given rate *optimal type-I shaping codes*. We develop a theoretical bound on the trade-off between the rate – or more precisely, the corresponding *expansion factor* – and the average cost of a type-I shaping

code (Theorem 6). We then study shaping codes that minimize total cost (minimum average cost per source symbol). We refer to the problem of minimizing the total cost as the *type-II shaping problem* and shaping codes that achieve the minimum total cost are referred to as *optimal type-II shaping codes*. We derive the relationship between the code expansion factor and the total cost and determine the optimal expansion factor (Theorem 7). We then prove an equivalence theorem showing that an optimal type-I shaping code can be realized using an optimal type-II shaping code for another suitably chosen costly channel model (Theorem 8). We can therefore solve the type-I shaping problem using known coding techniques such as generalized Shannon-Fano codes [14]. A consequence of the analysis is a separation theorem for type-II shaping codes, which states that optimal shaping can be achieved by a concatenation of lossless compression and optimal shaping for a uniform i.i.d. source. This provides an alternative architecture for implementing asymptotically optimal shaping codes using, for example, Varn codes. Finally, we prove a separation theorem for type-I shaping codes with given expansion factor, using a careful analysis of the behavior of the minimum average cost as a function of the expansion factor.

3.1.2 Distribution Matching Codes

We then systematically study the problem of designing optimal fixed-to-variable length, prefix-free DM codes from the perspective of word-valued sources and shaping codes. The degree of distribution matching is measured by the KL-divergence between the distribution on word-valued source output sequences and the distribution on those sequences generated i.i.d. according to the target distribution. Vanishing asymptotic normalized KL-divergence at the sequence level, suggested by the approach in [56] and also studied by Soriaga [68], is used as the criterion for optimality. We first characterize the expansion factor of an optimal DM code for a general i.i.d. source (Theorem 12). We then show that an optimal type-II shaping code for a cost model determined by the negative logarithm of a target distribution is an optimal DM code for that distribution (Theorem 13). (This “self-information” cost model was also used in [65] to

design information divergence optimal fixed-to-fixed DM codes using shell mapping.)

The connection between shaping codes and DM codes suggests another measure for evaluating DM code performance, which we refer to as *generalized expansion factor* (GEF). We establish a lower bound on the generalized expansion factor, and show that a code that achieves the lower bound is an optimal DM code (Theorem 15). This implies that Varn codes are asymptotically optimal DM codes for a uniform i.i.d. source. Using the GEF, we also extend the separation theorem of shaping codes to DM codes (Theorem 16).

Finally, we discuss relationships between different DM code performance measures. We show that for a DM code with fixed codebook size, minimizing the GEF is equivalent to minimizing the informational divergence introduced in [3], leading to the conclusion that Varn codes designed for the appropriate cost model minimize informational divergence (Theorem 17), generalizing a result for binary Tunstall codes in [3]. We also give an explicit description of the relationship between the normalized informational divergence of a DM code and its expansion factor (Theorem 18 and Remark 3.6.3), refining a bound in [6].

3.2 Information-theoretic Preliminaries

3.2.1 Basic Model

First, we fix some notation. Let $\mathbf{X} = X_1 X_2 \dots$, where $X_i \sim X$ for all i , be an i.i.d. source with alphabet $\mathcal{X} = \{\alpha_1, \dots, \alpha_u\}$. We use $|\mathcal{X}|$ to denote the size of the alphabet and use $P(x^*)$ to denote the probability of any finite sequence x^* . Let $\mathcal{Y} = \{\beta_1, \dots, \beta_v\}$ be an alphabet and $\mathcal{Y}^*$ be the set of all finite sequences over $\mathcal{Y}$, including the null string λ of length 0. Each β_i is associated with a cost C_i . Without loss of generality, we assume that $0 \leq C_1 \leq C_2 \leq \dots \leq C_v$, and we also assume that there exists at least one pair of costs, C_i and C_j , such that $C_i \neq C_j$. We use a cost vector $\mathcal{C} = [C_1, C_2, \dots, C_v]$ to represent the cost associated with alphabet $\mathcal{Y}$.

A general shaping code is defined as a prefix-free variable-length mapping $\phi : \mathcal{X}^q \rightarrow \mathcal{Y}^*$

which maps a length- q data word x_1^q to a variable-length codeword y^* . We use $\mathbf{Y}$ to denote the process $\phi(\mathbf{X}^q)$, where $\mathbf{X}^q$ is the vector process $X_1^q, X_{q+1}^{2q}, \dots$. The entropy rate of the process $\mathbf{Y}$ is

$$H(\mathbf{Y}) = \lim_{n \rightarrow \infty} \frac{1}{n} H(Y_1 Y_2 \dots Y_n). \quad (3.1)$$

We denote the length of a codeword $\phi(x_1^q)$ by $L(\phi(x_1^q))$ and the expected length of codewords generated by a sequence of length- q source words is given by

$$E(L) = \sum_{x_1^q \in \mathcal{X}^q} P(x_1^q) L(\phi(x_1^q)). \quad (3.2)$$

The *expansion factor* is defined as the ratio of the expected codeword length to the length of the input source word, namely

$$f = E[L]/q. \quad (3.3)$$

Remark 3.2.1. Endurance codes and direct shaping codes can be treated as special cases of this general class of shaping codes. Endurance codes are used when the source has a uniform i.i.d. distribution, with entropy rate $H(\mathbf{X}) = \log_2 |\mathcal{X}|$. A length- m direct shaping code is a shaping code with $q = 1$, $f = 1$, where both $\mathbf{X}$ and $\mathbf{Y}$ have alphabet size 2^m . $\square$

The pair $\mathbf{X}$ and ϕ form a word valued source, as defined in[56]. The following theorem, proved in[56], gives the entropy rate of the shaping code $\phi(\mathbf{X}^q)$.

Theorem 3.2.1. *For a prefix-free variable-length code $\mathbf{Y} = \phi(\mathbf{X}^q)$ such that $H(\mathbf{X}^q) < \infty$ and $E(L) < \infty$, the entropy rate of the encoder output satisfies*

$$H(\mathbf{Y}) = \frac{H(\mathbf{X}^q)}{E(L)} = \frac{qH(\mathbf{X})}{E(L)}. \quad (3.4)$$

$\square$

3.2.2 Asymptotic Symbol Occurrence Probability

For simplicity and without loss of generality, we assume $q = 1$. The mapping is $\phi : \mathcal{X} \rightarrow \mathcal{Y}^*$. Let y_1^l denote the first l symbols of $\phi(\mathbf{X})$. We assume the cost is independent and additive, so the cost of sequence y_1^l can be expressed as

$$W(y_1^l) = \sum_{i=1}^v N_i(y_1^l) C_i \quad (3.5)$$

where $N_i(y_1^l)$ stands for the number of occurrences of β_i in sequence y_1^l . The cost per code symbol is therefore $\sum_i N_i(y_1^l) C_i / l$. Let

$$Q(y_1^l) = \Pr\{Y_1^l = y_1^l\} \quad (3.6)$$

denote the probability distribution of Y_1^l . The expected cost per symbol of a length- l shaping code sequence is

$$\begin{aligned} W_l &= \sum_{y_1^l \in \mathcal{Y}^l} Q(y_1^l) W(y_1^l) / l \\ &= \sum_{i=1}^v \sum_{y_1^l \in \mathcal{Y}^l} Q(y_1^l) N_i(y_1^l) C_i / l. \end{aligned} \quad (3.7)$$

The asymptotic expected cost per symbol, or *average cost* of a shaping code is

$$A(\phi(\mathbf{X})) = \lim_{l \rightarrow \infty} W_l. \quad (3.8)$$

Let

$$\hat{P}_i = \lim_{l \rightarrow \infty} \sum_{y_1^l} Q(y_1^l) N_i(y_1^l) / l = \lim_{l \rightarrow \infty} \frac{E(N_i(Y_1^l))}{l} \quad (3.9)$$

be the asymptotic probability of occurrence of β_i . Then the average cost of a shaping code can be expressed as

$$A(\phi(\mathbf{X})) = \sum_i \hat{P}_i C_i. \quad (3.10)$$

In the rest of this subsection, we will show how to calculate $\hat{P}_i$. Define the prefix operator π as $y_1^n \pi^i = y_1^{n-i}$ for $0 \leq i < n$ and $y_1^n \pi^i = \lambda$ for $i \geq n$. Let $\pi\{y^*\}$ denote the set of all the prefixes of a sequence y^* . We denote by $\mathcal{G}_\phi(y_1^l)$ the set of all sequences $x^* \in \mathcal{X}^*$ such that y_1^l is a prefix of $\phi(x^*)$ but not of $\phi(x^* \pi)$. That is,

$$\mathcal{G}_\phi(y_1^l) = \{x^* \in \mathcal{X}^* | y_1^l \in \pi\{\phi(x^*)\} \wedge |\phi(x^* \pi)| < l\} \quad (3.11)$$

and the distribution of y_1^l can be expressed as

$$Q(y_1^l) = \sum_{x^* \in \mathcal{G}_\phi(y_1^l)} P(x^*). \quad (3.12)$$

We define by M_l the minimum length of a sequence x_1^n such that $|\phi(x_1^n)| \geq l$ and let S_{M_l} be the length of $\phi(x_1^{M_l})$. Note that

$$S_{M_l-1} < l \leq S_{M_l}. \quad (3.13)$$

According to [56], the random variable M_l satisfies the property of being a *stopping rule* for the sequence of i.i.d. random variables $\{\phi(X^\infty)\}$. Wald's equality [80] then implies that

$$E(N_i(\phi(X_1^{M_l}))) = E(N_i(\phi(X)))E(M_l). \quad (3.14)$$

The following two lemmas were proved in [56].

Lemma 3.2.2. *Given a nonnegative-valued function f , let $F_i = f(X_i)$. If $E(F) < \infty$, then*

$$\lim_{l \rightarrow \infty} \frac{E(F_{M_l})}{l} = 0. \quad (3.15)$$

□

Remark 3.2.2. The previous lemma is not obvious, because even when $E(F) < \infty$, $E(F_{M_l})$ is not necessarily equal to $E(F)$. □

Lemma 3.2.3. *If $E[L] < \infty$, then*

$$\lim_{l \rightarrow \infty} \frac{E[M_l]}{l} = \frac{1}{E(L)}. \quad (3.16)$$

□

Using these results, we derive a lemma which tells us how to calculate the asymptotic occurrence probability of the encoder output process $\mathbf{Y}$.

Lemma 3.2.4. *For a prefix-free variable-length code $\phi : X^q \rightarrow \mathcal{Y}^*$ such that $E(N_i(\phi(X^q))) < \infty$ for all symbols β_i and $E(L) < \infty$, the asymptotic probability of occurrence $\hat{P}_i$ of β_i is given by*

$$\hat{P}_i = E(N_i(\phi(X^q))) \frac{1}{E(L)}. \quad (3.17)$$

Proof. In this proof, without loss of generality, we will assume $q = 1$. First, we evaluate the expectation of the sequence of random variable $\{N_i(\phi(X_1^{M_l}))\}_{l=1}^{\infty}$. Combining Lemma 3.2.3 with equation (3.14), we have

$$\begin{aligned} \lim_{l \rightarrow \infty} \frac{1}{l} E(N_i(\phi(X_1^{M_l}))) &= \lim_{l \rightarrow \infty} \frac{1}{l} E(N_i(\phi(X))) E(M_l) \\ &= E(N_i(\phi(X))) \lim_{l \rightarrow \infty} \frac{1}{l} E(M_l) \\ &= E(N_i(\phi(X))) \frac{1}{E(L)}. \end{aligned} \quad (3.18)$$

Similarly, we have

$$\begin{aligned} \lim_{l \rightarrow \infty} \frac{1}{l} E(N_i(\phi(X_1^{M_l-1}))) &= \lim_{l \rightarrow \infty} \frac{1}{l} E(N_i(\phi(X_1^{M_l})) - N_i(\phi(X_{M_l}))) \\ &= E(N_i(\phi(X))) \frac{1}{E(L)} - \lim_{l \rightarrow \infty} \frac{1}{l} E(N_i(\phi(X_{M_l}))) \\ &= E(N_i(\phi(X))) \frac{1}{E(L)}, \end{aligned} \quad (3.19)$$

where $\lim_{l \rightarrow \infty} \frac{1}{l} E(N_i(\phi(X_{M_l}))) = 0$ follows from Lemma 3.2.2.

By definition,

$$E(N_i(\phi(X_1^{M_l}))) = \sum_{y_1^l} \sum_{x_1^{M_l} \in \mathcal{G}_\phi(y_1^l)} P(x_1^{M_l}) N_i(\phi(x_1^{M_l})). \quad (3.20)$$

Since $S_{M_l} > l$, we have $N_i(\phi(x_1^{M_l})) \geq N_i(y_1^l)$ and $E(N_i(y_1^l))$ can be bounded as follows

$$\begin{aligned} E(N_i(Y_1^l)) &= \sum_{y_1^l} N_i(y_1^l) Q(y_1^l) = \sum_{y_1^l} N_i(y_1^l) \sum_{x_1^{M_l} \in \mathcal{G}_\phi(y_1^l)} P(x_1^{M_l}) \\ &\leq \sum_{y_1^l} N_i(\phi(x_1^{M_l})) \sum_{x_1^{M_l} \in \mathcal{G}_\phi(y_1^l)} P(x_1^{M_l}) = \sum_{y_1^l} \sum_{x_1^{M_l} \in \mathcal{G}_\phi(y_1^l)} P(x_1^{M_l}) N_i(\phi(x_1^{M_l})) \\ &= E(N_i(\phi(X_1^{M_l}))). \end{aligned} \quad (3.21)$$

Similarly, $N_i(\phi(x_1^{M_l-1})) \leq N_i(\phi(y_1^l))$ and $E(N_i(Y_1^l))$ is lower bounded by

$$\begin{aligned} E(N_i(Y_1^l)) &= \sum_{y_1^l} N_i(y_1^l) \sum_{x_1^{M_l} \in \mathcal{G}_\phi(y_1^l)} P(x_1^{M_l}) \\ &\geq \sum_{y_1^l} \sum_{x_1^{M_l} \in \mathcal{G}_\phi(y_1^l)} P(x_1^{M_l}) N_i(\phi(x_1^{M_l-1})) \\ &= E(N_i(\phi(X_1^{M_l-1}))). \end{aligned} \quad (3.22)$$

Equations (3.21) and (3.22) imply that

$$\begin{aligned} \limsup_{l \rightarrow \infty} \frac{1}{l} E(N_i(Y_1^l)) &\leq \liminf_{l \rightarrow \infty} \frac{1}{l} E(N_i(\phi(X_1^{M_l}))) \\ &= E(N_i(\phi(X))) \frac{1}{E(L)} \end{aligned} \quad (3.23)$$

and

$$\begin{aligned} \liminf_{l \rightarrow \infty} \frac{1}{l} E(N_i(Y_1^l)) &\geq \limsup_{l \rightarrow \infty} \frac{1}{l} E(N_i(\phi(X_1^{M_l-1}))) \\ &= E(N_i(\phi(X))) \frac{1}{E(L)}. \end{aligned} \quad (3.24)$$

Thus we conclude that

$$\hat{P}_i = \lim_{l \rightarrow \infty} \frac{1}{l} E(N_i(Y_1^l)) = E(N_i(\phi(X))) \frac{1}{E(L)}. \quad (3.25)$$

□

It is easy to check that $\sum_i \hat{P}_i = 1$, so this distribution is well defined.

3.2.3 Lower Bound on Symbol Distribution Entropy

Consider a prefix-free variable-length code ϕ as in Lemma 3.2.4. Let $\hat{Y}_1^l$ denote an i.i.d. sequence of length l and with distribution $\{\hat{P}_i\}$. The probability of a length- l sequence y_1^l with respect to this distribution is $\hat{P}(y_1^l) = \prod_i \hat{P}_i^{N_i(y_1^l)}$. The Kullback-Leibler (KL) divergence (also known as the KL-distance or relative entropy) [12] is a measure of the inefficiency caused by an approximation. The KL-divergence between Y_1^l and $\hat{Y}_1^l$ is

$$D(Y_1^l || \hat{Y}_1^l) = \sum_{y_1^l \in \mathcal{Y}^l} Q(y_1^l) \log_2 \frac{Q(y_1^l)}{\hat{P}(y_1^l)}. \quad (3.26)$$

The following lemma provides a lower bound on the symbol distribution entropy.

Lemma 3.2.5. *The asymptotic normalized KL-divergence satisfies*

$$\lim_{l \rightarrow \infty} \frac{1}{l} D(Y_1^l || \hat{Y}_1^l) = H(\hat{Y}) - H(\mathbf{Y}), \quad (3.27)$$

where $H(\hat{Y}) = -\sum_i \hat{P}_i \log_2 \hat{P}_i$. Specifically,

$$H(\hat{Y}) \geq H(\mathbf{Y}). \quad (3.28)$$

Proof. We rewrite the $D(Y_1^l || \hat{Y}_1^l)$ as

$$\begin{aligned} D(Y_1^l || \hat{Y}_1^l) &= \sum_{y_1^l \in \mathcal{Y}^l} Q(y_1^l) \log_2 \frac{Q(y_1^l)}{\hat{P}(y_1^l)} \\ &= \sum_{y_1^l \in \mathcal{Y}^l} Q(y_1^l) \log_2 Q(y_1^l) - \sum_{y_1^l \in \mathcal{Y}^l} Q(y_1^l) \log_2 \hat{P}(y_1^l) \\ &= -H(Y_1^l) - \sum_{y_1^l \in \mathcal{Y}^l} Q(y_1^l) \log_2 \hat{P}(y_1^l). \end{aligned} \quad (3.29)$$

The second term of the right-hand side of this equation is

$$\begin{aligned} \sum_{y_1^l \in \mathcal{Y}^l} Q(y_1^l) \log_2 \hat{P}(y_1^l) &= \sum_{y_1^l \in \mathcal{Y}^l} Q(y_1^l) \log_2 \prod_i \hat{P}_i^{N_i(y_1^l)} \\ &= \sum_{y_1^l \in \mathcal{Y}^l} Q(y_1^l) \sum_i N_i(y_1^l) \log_2 \hat{P}_i \\ &= \sum_i \log_2 \hat{P}_i \sum_{y_1^l \in \mathcal{Y}^l} Q(y_1^l) N_i(y_1^l). \end{aligned} \quad (3.30)$$

Combining equations (3.29) and (3.30), we have

$$\begin{aligned} &\lim_{l \rightarrow \infty} \frac{1}{l} D(Y_1^l || \hat{Y}_1^l) \\ &= -\lim_{l \rightarrow \infty} \frac{1}{l} H(Y_1^l) - \sum_i \log_2 \hat{P}_i \lim_{l \rightarrow \infty} \sum_{y_1^l \in \mathcal{Y}^l} \frac{Q(y_1^l) N_i(y_1^l)}{l} \\ &= -H(\mathbf{Y}) - \sum_i \hat{P}_i \log_2 \hat{P}_i = H(\hat{Y}) - H(\mathbf{Y}). \end{aligned} \quad (3.31)$$

Using the fact that $D(Y_1^l || \hat{Y}_1^l) \geq 0$, we have

$$H(\hat{Y}) \geq H(\mathbf{Y}). \quad (3.32)$$

This completes the proof. $\square$

Remark 3.2.3. We note that the entropy rate bound (3.32) also follows from applying the independence bound for joint entropy of a random vector, invoking concavity of the entropy function, and taking the limit in the length of the vector. $\square$

Remark 3.2.4. From the proof, we see that $H(\hat{Y}) = H(\mathbf{Y})$ implies $\lim_{l \rightarrow \infty} \frac{1}{l} D(Y^l || \hat{Y}^l) = 0$. Therefore, the codeword sequence $Y_1 Y_2 \cdots$ approximates an i.i.d. sequence generated by $\hat{Y}$. $\square$

Example 3.2.1. Consider a uniform i.i.d. binary source $\mathbf{X}$ and a prefix-free variable-length code defined by the mapping $\{00 \rightarrow 000, 01 \rightarrow 001, 10 \rightarrow 01, 11 \rightarrow 1\}$. The occurrence probabilities of symbols 0 and 1 are $2/3$ and $1/3$, respectively. The symbol distribution entropy is

$$H(\hat{Y}) = -\frac{1}{3} \log_2 \frac{1}{3} - \frac{2}{3} \log_2 \frac{2}{3} \simeq 0.9183. \quad (3.33)$$

The entropy rate of the shaping code sequence is

$$H(\mathbf{Y}) = \frac{H(\mathbf{X}^2)}{E(L)} = \frac{2}{2.25} = 0.8889. \quad (3.34)$$

We see that, in this case, $H(\mathbf{Y}) < H(\hat{Y})$. $\square$

3.3 Optimal Shaping Codes

3.3.1 Cost Minimizing Probability Distribution

In this subsection, we discuss the properties of optimal shaping codes. We consider two scenarios. First, we analyze shaping codes that minimize the average cost with a given expansion factor. We then analyze shaping codes that minimize the expected cost per source symbol, or total cost.

We refer to the first minimization problem as the *type-I shaping problem*, and we call shaping codes that achieve the minimum average cost for a given expansion factor *optimal type-I shaping codes*. The following theorem gives a lower bound on the average cost and the corresponding asymptotic symbol occurrence probabilities.

Theorem 3.3.1. *Given the source $\mathbf{X}$ and cost vector $\mathbf{C}$, the average cost of a type-I shaping code $\phi : \mathcal{X}^q \rightarrow \mathcal{Y}^*$ with expansion factor f is lower bounded by $\sum_i \hat{P}_i C_i$, with*

$$\hat{P}_i = \frac{1}{N} 2^{-\mu C_i}, \quad (3.35)$$

where N is a normalization constant such that $\sum_i \hat{P}_i = 1$ and μ is a non-negative constant such that $\sum_i -\hat{P}_i \log \hat{P}_i = H(\mathbf{X})/f$.

Proof. From Theorem 3.2.1 and Lemma 3.2.5, we see that, for a shaping code ϕ with expansion factor f , the following inequality holds:

$$H(\hat{\mathbf{Y}}) \geq H(\mathbf{Y}) = \frac{qH(\mathbf{X})}{E(L)} = \frac{H(\mathbf{X})}{f}. \quad (3.36)$$

To calculate the minimum possible average cost, we must solve the optimization problem:

$$\begin{aligned}
& \underset{\hat{P}_i}{\text{minimize}} && \sum_i \hat{P}_i C_i \\
& \text{subject to} && H(\hat{Y}) \geq \frac{H(\mathbf{X})}{f} \\
& && \sum_i \hat{P}_i = 1.
\end{aligned} \tag{3.37}$$

We divide this optimization problem into two parts. First, we fix $H(\hat{Y})$ and find the optimal symbol occurrence probabilities. Then we find the optimal $H(\hat{Y})$ to minimize the average cost. The optimization problem then becomes

$$\begin{aligned}
& \underset{H(\hat{Y})}{\text{minimize}} \quad \underset{\hat{P}_i}{\text{minimize}} && \sum_i \hat{P}_i C_i \\
& \text{subject to} && H(\hat{Y}) \geq \frac{H(\mathbf{X})}{f} \\
& && \sum_i \hat{P}_i = 1.
\end{aligned} \tag{3.38}$$

If we fix $H(\hat{Y})$, we can solve the optimization problem by using the method of Lagrange multipliers. The solution is

$$\hat{P}_i = \frac{1}{N} 2^{-\mu C_i} \tag{3.39}$$

where $N = \sum_i 2^{-\mu C_i}$ is a normalization constant and μ is a non-negative constant such that

$$H(\hat{Y}) = \sum_i -\hat{P}_i \log_2 \hat{P}_i. \tag{3.40}$$

Note that $\mu = 0$ if and only if $H(\hat{Y}) = \log_2 |\mathcal{Y}|$. For simplicity, let h denote $H(\hat{Y})$. Then μ and N are functions of h , which we denote by $N \stackrel{\text{def}}{=} N(h)$ and $\mu \stackrel{\text{def}}{=} \mu(h)$, respectively. Let $C(h) = \sum_i \frac{C_i}{N(h)} 2^{-\mu(h) C_i}$ be the minimum cost, given that $h \geq \frac{H(\mathbf{X})}{f}$. From (3.40), we see that

$$C(h) = \frac{h - \log_2 N}{\mu} \quad \text{when } \mu > 0. \tag{3.41}$$

The optimization problem we have reduced to here, minimizing the average cost of a probability

mass function subject to a lower bound on entropy, is dual to the problem considered in prior work such as [51, Problem 1.8] and [5, Sec. 5.2], which is a special case of results in [52], [32], and [35]. The relationship between entropy rate and average cost discussed in these papers has the same functional form as (3.41). We can apply the analysis in [5, Sec. 5.2], to conclude that

$$\frac{dh}{dC} = \mu \Rightarrow \frac{dC}{dh} > 0 \quad \text{when } \mu > 0. \quad (3.42)$$

Therefore, the minimum cost for a shaping code with expansion factor f is achieved when $h = H(\hat{Y}) = \frac{H(X)}{f}$.

Note that we have minimized average cost by optimizing the asymptotic symbol occurrence probability $\hat{P}_i$ of a prefix-free variable-length mapping whose output entropy rate is fixed, without consideration of whether the output sequence coincides with an i.i.d. sequence. $\square$

Remark 3.3.1. If the source $\mathbf{X}$ has a uniform distribution, then μ satisfies $-f \sum_i \hat{P}_i \log \hat{P}_i = \log_2 |\mathcal{X}|$. Thus, we recover the result in [31] characterizing endurance codes with minimum average cost. $\square$

Remark 3.3.2. When the minimum average cost is achieved, we have $H(\hat{Y}) = H(\mathbf{Y})$. Thus, the codeword sequence approximates an i.i.d. sequence generated by distribution $\{\hat{P}_i\}$ (see Remark 3.2.4). $\square$

Given a prefix-free variable-length shaping code $\phi : \mathcal{X}^q \rightarrow \mathcal{Y}^*$, assume that after nq source symbols are encoded, the codeword sequence is $\phi(x_1^{nq})$. As in equations (3.7), (3.8), (3.9), we formally define the expected cost per source symbol, or *total cost* of a shaping code as

$$\begin{aligned} T(\phi(\mathbf{X}^q)) &= \frac{\sum_i E(N_i(\phi(X^{nq})))C_i}{nq} = \frac{\sum_i E(N_i(\phi(X^q)))C_i}{q} \\ &= \frac{E(L)}{q} \frac{\sum_i E(N_i(\phi(X^q)))C_i}{E(L)} = f \sum_{i=1}^v \hat{P}_i C_i. \end{aligned} \quad (3.43)$$

We refer to the problem of minimizing the total cost as the *type-II shaping problem*.

Shaping codes that achieve the minimum total cost are referred to as *optimal type-II shaping codes*. The corresponding optimization problem is as follows:

$$\begin{aligned}
& \underset{\hat{P}_i, f}{\text{minimize}} && f \sum_{i=1}^v \hat{P}_i C_i \\
& \text{subject to} && H(\hat{Y}) \geq H(\mathbf{Y}) = \frac{H(\mathbf{X})}{f} \\
& && \sum_i \hat{P}_i = 1.
\end{aligned} \tag{3.44}$$

Using Theorem 3.3.1, we can calculate the total cost as a function of the expansion factor f . Fig. 3.1 shows the total cost curve for a quaternary source and code alphabet, a uniformly distributed source $\mathbf{X}$, and cost vector $\mathcal{C} = [1, 2, 3, 4]$. There is an optimal value of f and corresponding minimum total cost.

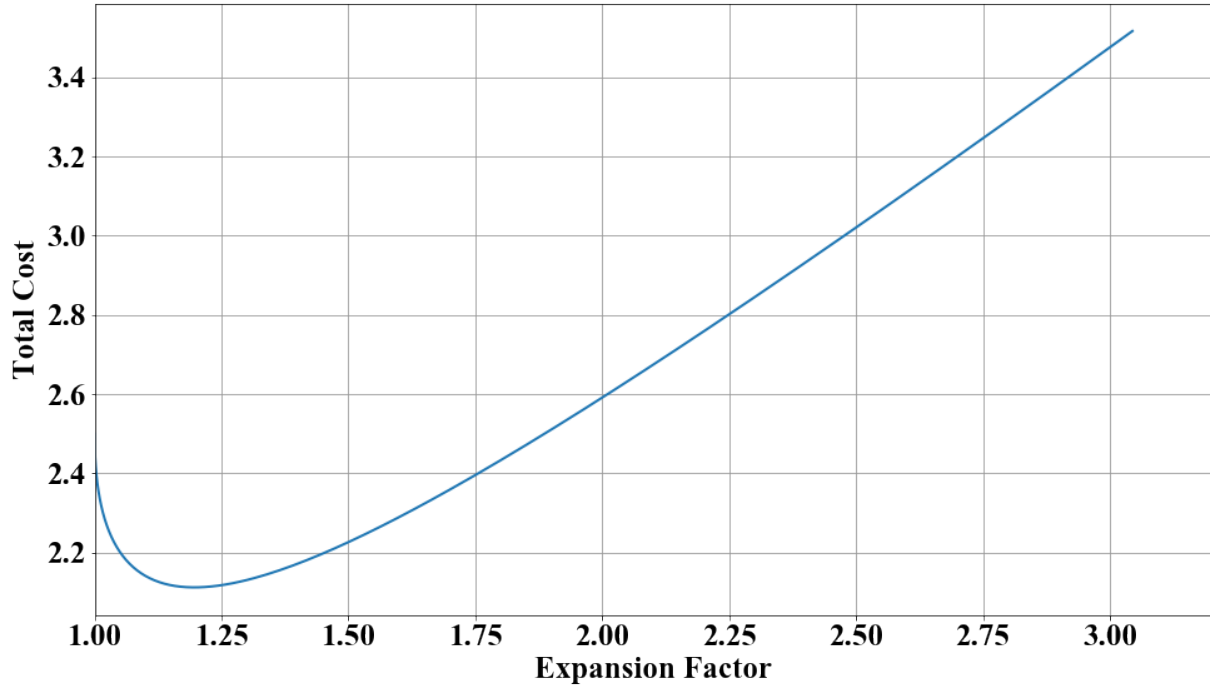


Figure 3.1: Total cost versus f for random source with $\mathcal{C} = [1, 2, 3, 4]$.

We now determine the minimum achievable total cost of a shaping code.

Theorem 3.3.2. *Given the source $\mathbf{X}$ and cost vector $\mathbf{C}$, if $C_1 \neq 0$, then the minimum total cost of a shaping code $\phi : X^q \rightarrow \mathcal{Y}^*$ is given by $f \sum_i \hat{P}_i C_i$, where $\hat{P}_i = 2^{-\mu C_i}$, μ is a positive constant selected such that $\sum_i 2^{-\mu C_i} = 1$. The corresponding expansion factor f is*

$$f = \frac{H(\mathbf{X})}{-\sum_i \hat{P}_i \log_2 \hat{P}_i}. \quad (3.45)$$

If $C_1 = 0$, then the total cost is a decreasing function of f .

Proof. We solve the optimization problem:

$$\begin{aligned} & \underset{\hat{P}_i, f}{\text{minimize}} && f \sum_i \hat{P}_i C_i \\ & \text{subject to} && H(\hat{Y}) \geq \frac{H(\mathbf{X})}{f} \\ & && \sum_i \hat{P}_i = 1. \end{aligned} \quad (3.46)$$

We divide this optimization problem into two parts. First we fix expansion factor f and find the minimum achievable average cost using Theorem 3.3.1. Then we find the optimal f to minimize the total cost. The optimization problem then becomes:

$$\begin{aligned} & \underset{f}{\text{minimize}} && \underset{\hat{P}_i}{\text{minimize}} && \sum_i f \hat{P}_i C_i \\ & \text{subject to} && && H(\hat{Y}) \geq \frac{H(\mathbf{X})}{f} \\ & && && \sum_i \hat{P}_i = 1. \end{aligned} \quad (3.47)$$

For fixed f , the symbol occurrence probability corresponding to the minimum average cost is

$$P'_i = \frac{1}{N} 2^{-\mu C_i}, \quad (3.48)$$

where μ is a constant such that

$$\frac{1}{f} = \frac{-\sum_i P'_i \log_2 P'_i}{\log_2 |\mathcal{X}|} \quad (3.49)$$

and N is the normalization factor

$$N = \sum_i 2^{-\mu C_i}. \quad (3.50)$$

Treating f as a function of μ , and calculating $df/d\mu$, we see that $f(\mu)$ is a monotone increasing function of μ , with $f(0) = \frac{H(\mathbf{X})}{\log_2 |\mathcal{Y}|}$. Specifically, replacing the P'_i in (3.49) by (3.48) gives

$$f = \frac{NH(\mathbf{X})}{(\sum_i \mu C_i 2^{-\mu C_i}) + N \log_2 N}. \quad (3.51)$$

The derivative of f with respect to μ is

$$\frac{df}{d\mu} = \frac{\mu \ln 2 H(\mathbf{X}) \sum_{i < j} 2^{-\mu(C_i + C_j)} (C_i - C_j)^2}{[(\sum_i \mu C_i 2^{-\mu C_i}) + N \log_2 N]^2}, \quad (3.52)$$

which is easily seen to be positive for $\mu > 0$. The optimization problem is then equivalent to:

$$\begin{aligned} \text{minimize} \quad & T(\mu) = H(\mathbf{X}) \frac{\sum_i C_i 2^{-\mu C_i}}{\sum_i \mu C_i 2^{-\mu C_i} + N \log_2 N} \\ \text{subject to} \quad & \mu \geq 0. \end{aligned} \quad (3.53)$$

Calculating $dT/d\mu$, we see that its sign is the negative of the sign of $\log_2 N$. Specifically, we find that

$$\frac{dT}{d\mu} = -\ln 2 H(\mathbf{X}) \log_2 N \frac{N \sum_i C_i^2 2^{-\mu C_i} - (\sum_i C_i 2^{-\mu C_i})^2}{(\sum_i \mu C_i 2^{-\mu C_i} + N \log_2 N)^2}. \quad (3.54)$$

Observe that

$$\begin{aligned} & N \sum_i C_i^2 2^{-\mu C_i} - (\sum_i C_i 2^{-\mu C_i})^2 \\ &= \sum_{i,j} C_i^2 2^{-\mu(C_i + C_j)} - \sum_i C_i^2 2^{-2\mu C_i} - \sum_{i < j} 2C_i C_j 2^{-\mu(C_i + C_j)} \\ &= \sum_{i < j} (C_i^2 + C_j^2 - 2C_i C_j) 2^{-\mu(C_i + C_j)} \\ &= \sum_{i < j} (C_i - C_j)^2 2^{-\mu(C_i + C_j)} > 0. \end{aligned} \quad (3.55)$$

Therefore,

$$\frac{dT}{d\mu} = -\ln 2 H(\mathbf{X}) \log_2 N \frac{\sum_{i < j} (C_i - C_j)^2 2^{-\mu(C_i + C_j)}}{(\sum_i \mu C_i 2^{-\mu C_i} + N \log_2 N)^2}. \quad (3.56)$$

The claimed relationship between the sign of $dT/d\mu$ and the sign of $\log_2 N$ is then evident.

It follows that if $C_1 = 0$, then $N = \sum_i 2^{-\mu C_i} > 1$, implying that $T(\mu)$ is a monotone decreasing function on $[0, \infty)$. On the other hand, if $C_1 > 0$, then when $\mu = 0$, we have $N = |\mathcal{Y}|$, so $\log_2 N > 0$. Since $dN/d\mu < 0$ for all μ , we conclude that T will decrease as μ increases, reaching a minimum at $N = 1$. Beyond that point, $dN/d\mu > 0$.

Thus, the corresponding expansion factor that achieves the minimum total cost is

$$f = \frac{H(\mathbf{X})}{-\sum_i \hat{P}_i \log_2 \hat{P}_i} \quad (3.57)$$

where $\hat{P}_i = 2^{-\mu C_i}$, and μ is a positive constant satisfying $\sum_i 2^{-\mu C_i} = 1$. $\square$

Remark 3.3.3. For a positive cost vector C , the minimum achievable total cost is

$$\begin{aligned} T(\phi(\mathbf{X}^q)) &= f \sum_i \hat{P}_i C_i = \frac{H(\mathbf{X})}{-\sum_i \hat{P}_i \log_2 \hat{P}_i} \sum_i \hat{P}_i C_i \\ &= \frac{H(\mathbf{X})}{-\sum_i \hat{P}_i \log_2 2^{-\mu C_i}} \sum_i \hat{P}_i C_i = \frac{H(\mathbf{X})}{\mu \sum_i \hat{P}_i C_i} \sum_i \hat{P}_i C_i \\ &= \frac{H(\mathbf{X})}{\mu}. \end{aligned} \quad (3.58)$$

The capacity of a noiseless finite-state costly channel, which is essentially the inverse of the minimum total cost, was considered in [52], [32], [35], [9], and [5, Sec 6] from combinatorial and probabilistic perspectives. Equivalences between the combinatorial and probabilistic definitions of capacity were established, extending the original results of Shannon.

In [14, Theorem 4.4] and [26, Theorem 1], the minimum total cost of a prefix-free variable-length code was determined. In [5, Section 4], an asymptotically optimal variable-to-fixed length code for binary uniform i.i.d. source that minimizes total cost was proposed. However, these

works did not address the code expansion factor and asymptotic symbol occurrence probability corresponding to the minimum total cost.

In [51, Problems 1.8], [49] and [5, Sec. 5.2], the relationship between the maximum entropy of a probability mass function on an alphabet with cost subject to an average cost constraint was discussed. However, these works did not explore the functional relationship between the total cost and the expansion factor of a code. Here, using the word-valued source perspective, we establish the relationship between the total cost of a rate-constrained prefix-free code and its expansion factor. This relationship plays an important role in the proof of the separation theorem (Theorem 3.3.6) in Appendix 3.3.6. We also address the special case of zero lowest cost, i.e., $C_1 = 0$, in which no global minimum can be reached. $\square$

Remark 3.3.4. If we only apply optimal lossless compression to the source $\mathbf{X}$, the code sequence has a uniform distribution. Therefore, we have $\mu = 0$ and $N = |\mathcal{Y}| > 1$. This implies that simply applying compression to the source data is not the best way to reduce the total cost. $\square$

3.3.2 Optimal Data Shaping Code Design

Many previous works investigated type-II shaping code design. For example, see [34], [78], and [14]. In this subsection, we consider the problem of designing an optimal type-I shaping code by transforming this problem into a type-II shaping problem. Combining Theorems 3.3.1 and 3.3.2, we can prove the following equivalence theorem.

Theorem 3.3.3. *A code that achieves the minimum total cost for cost vector C' also achieves minimum average cost for cost vector C and expansion factor f if*

$$C'_i = -\log_2 \hat{P}_i, \quad (3.59)$$

where $\{\hat{P}_i\}$ are the probabilities minimizing average cost for the cost vector C and expansion factor f .

Proof. First we consider the optimal type-II shaping code $\phi : \mathcal{X}^q \rightarrow \mathcal{Y}^*$ with cost vector $\mathcal{C}'$. By Theorem 3.3.2, this code generates codeword sequence with probability of occurrence $P'_i = 2^{-\mu C'_i}$, where μ satisfies the equation

$$\sum_i 2^{-\mu C'_i} = 1. \quad (3.60)$$

Since $C'_i = -\log_2 \hat{P}_i$, it is easy to check that the solution of equation (3.60) is $\mu = 1$. This means when the minimum total cost is achieved, the probability of occurrence of codeword sequence is $P'_i = 2^{-C'_i} = \hat{P}_i$ and the expansion factor of this code is

$$f' = \frac{H(X)}{-\sum_i P'_i \log_2 P'_i} = f \quad (3.61)$$

Referring to Theorem 3.3.1, we see that ϕ is also optimal with respect to minimizing average cost with cost vector $\mathcal{C}$ and expansion factor f . $\square$

When designing a type-I shaping code with expansion factor f and cost vector $\mathcal{C}$, we can first calculate the desired distribution $\{\hat{P}_i\}$, then transform this problem into a type-II shaping code problem for the channel with symbol cost $\{C'_i = -\log_2 \hat{P}_i\}$. Thus we can apply known type-II shaping code algorithms to solve this problem.

Remark 3.3.5. For an arbitrary i.i.d. source and a positive cost vector $\mathcal{C}$, generalized Shannon-Fano codes [14, Theorem 4.4] are tree-based variable-length codes, $\phi : \mathcal{X}^q \rightarrow \mathcal{Y}^*$, whose total cost is upper bounded by

$$\begin{aligned} T(\phi) &< \frac{H(\mathbf{X})}{\mu} + \frac{\max_i \{C_i\}}{q} \\ &\rightarrow \frac{H(\mathbf{X})}{\mu} \quad \text{as } q \rightarrow \infty. \end{aligned} \quad (3.62)$$

This coding scheme includes dividing the $[0, 1]$ interval based on $P(x_1^q)$ and calculating $2^{-\mu W}$, where W is the cost of a codeword. This construction may become impractical when q is large. $\square$

Remark 3.3.6. For a uniform i.i.d. source and a positive cost vector $\mathcal{C}$, Varn codes [78] are

tree-based, variable-length codes $\phi_K : \mathcal{X}^{\log_{|\mathcal{X}|} K} \rightarrow \mathcal{Y}^*$ that minimize total cost for a specified codebook size K . In [61], bounds were established on the average *codeword* cost for the Varn code with codebook size K , denoted $C(\phi_K)$. Specifically,

$$\frac{\log_2 K}{\mu} \leq C(\phi_K) \leq \frac{\log_2 K}{\mu} + \max_i \{C_i\}. \quad (3.63)$$

Dividing by $\log_{|\mathcal{X}|} K$, we see that the total cost of the Varn code with codebook size K is bounded by

$$\frac{\log_2 |\mathcal{X}|}{\mu} \leq T(\phi_K) \leq \frac{\log_2 |\mathcal{X}|}{\mu} + \frac{\max_i \{C_i\}}{\log_{|\mathcal{X}|} K}. \quad (3.64)$$

Therefore

$$\lim_{K \rightarrow \infty} T(\phi_K) = \frac{\log_2 |\mathcal{X}|}{\mu} \quad (3.65)$$

which implies that Varn codes are asymptotically optimal type-II shaping codes (see Remark 3.3.3). □

We now present a separation theorem for type-II shaping codes. It states that minimum total cost can be achieved by a concatenation of optimal lossless compression with an optimal type-II shaping code for a uniform i.i.d. source. The proof uses a construction based on typical sequences.

Theorem 3.3.4. *Given the source $\mathbf{X}$ and cost vector $\mathcal{C}$, the minimum total cost can be achieved by a concatenation of an optimal lossless compression code with an optimal type-II shaping code for a uniform i.i.d. source.*

Proof. Before proving the separation theorem, we first design a type-II shaping code for the uniform source. The *average codeword cost* $C(K)$ for a Varn code with codebook size K is bounded by

$$\log_2 K / \mu \leq C(K) \leq \log_2 K / \mu + \max_i C_i \quad (3.66)$$

where μ is a constant such that

$$\sum_i 2^{-\mu C_i} = 1. \quad (3.67)$$

Even though the average codeword cost is bounded, it is not guaranteed that the cost of every codeword satisfies the same bound. In order to ensure this, we instead use a modified design which we refer to as a modified Varn code. Given the binary alphabets $\{0, 1\}$ and $\mathcal{Y}$, a tree-based variable-length modified Varn code $\phi : \mathcal{X}^K \rightarrow \mathcal{Y}^*$ is designed as follows:

- Let $v = [(2^K - 1) \bmod (|\mathcal{Y}| - 1)]$.
- If $v > 0$, let $\delta = |\mathcal{Y}| - 1 - v$. Else if $v = 0$, let $\delta = 0$. Set $M = 2^K + \delta$.
- Design an exhaustive Varn code with codebook size M .
- Trim down the tree by getting rid of the δ branches with largest cost.

We note that $\delta \leq |\mathcal{Y}| - 2$.

The following lemma gives an upper bound on the codeword cost in a modified Varn code.

Lemma 3.3.5. *Every codeword of the modified Varn code has cost upper bounded by*

$$W_i \leq \log_2 M / \mu + \max_i C_i. \quad (3.68)$$

□

Proof. Consider the internal node that was the last one to be expanded and suppose it has cost W_0 . The cost of any leaf node is larger than W_0 , so

$$W_i \geq W_0. \quad (3.69)$$

Since this internal node is the last one to be expanded, its cost is larger than that of any other

internal node, implying

$$W_i \leq W_0 + \max C_i. \quad (3.70)$$

Since the tree is full, it is easy to check that

$$\sum_i 2^{-\mu W_i} = 1. \quad (3.71)$$

Combining equations (3.69) and (3.71), we have

$$1 = \sum_i 2^{-\mu W_i} \leq M 2^{-\mu W_0}. \quad (3.72)$$

This implies

$$W_0 \leq \log_2 M / \mu \quad (3.73)$$

and

$$W_i \leq W_0 + \max C_i \leq \log_2 M / \mu + \max C_i. \quad (3.74)$$

□

Now we are ready to prove the separation theorem. Let's first define a constant

$$D = \left\lceil \frac{\log_2 |\mathcal{X}| + 2}{H(\mathbf{X})} \right\rceil. \quad (3.75)$$

For any given $\gamma > 0$, define

$$\varepsilon = \min\left\{\frac{\mu\gamma}{2H(\mathbf{X})D}, \frac{1}{D}, \frac{\mu\gamma}{16}\right\} \quad (3.76)$$

and $\varepsilon' = \frac{\mu\gamma}{16}$. Consider the typical set $A_\varepsilon^{(q)}$ with respect to an i.i.d. source with entropy $H(\mathbf{X})$. There exists positive integer Q_1 such that when $q > Q_1$, $\Pr\{A_\varepsilon^{(q)}\} > 1 - \varepsilon$. There are $\leq 2^{q(H(\mathbf{X}) + \varepsilon)}$ length- q sequences x^q in $A_\varepsilon^{(q)}$, so we can use no more than $\lceil q(H + \varepsilon) + 1 \rceil$ bits to index them. We prefix all these sequences by a 0, giving a total length of $\lceil q(H(\mathbf{X}) + \varepsilon) + 2 \rceil$ to represent each

sequence in $A_\epsilon^{(q)}$. Similarly, we can index each sequence not in $A_\epsilon^{(q)}$ by using $\lceil q \log |\mathcal{X}| \rceil$ bits.

Prefixing these indices by 1, we have a prefix-free code ψ_1 for all sequences in $\mathcal{X}^q$.

Now we construct a length- $\lceil q(H(\mathbf{X}) + \epsilon) + 2 \rceil$ modified Varn code $\psi_2 : \mathcal{X}^{\lceil q(H(\mathbf{X}) + \epsilon) + 2 \rceil} \rightarrow \mathcal{Y}^*$. We use this code to encode the codeword sequence generated by ψ_1 . For every codeword in $A_\epsilon^{(q)}$, the cost is upper bounded by $\log_2 M/\mu + \max_i C_i$. For every codeword in the complement of $A_\epsilon^{(q)}$, which we denote by $B_\epsilon^{(q)}$, $\left\lceil \frac{\lceil q \log |\mathcal{X}| + 1 \rceil}{\lceil q(H(\mathbf{X}) + \epsilon) + 2 \rceil} \right\rceil$ codewords in ψ_2 are needed. The total cost is upper bounded by $(\left\lceil \frac{\lceil q \log |\mathcal{X}| + 1 \rceil}{\lceil q(H(\mathbf{X}) + \epsilon) + 2 \rceil} \right\rceil)(\log_2 M/\mu + \max_i C_i)$, where

$$\left\lceil \frac{\lceil q \log |\mathcal{X}| + 1 \rceil}{\lceil q(H(\mathbf{X}) + \epsilon) + 2 \rceil} \right\rceil \leq \left\lceil \frac{q \log |\mathcal{X}| + 2q}{qH(\mathbf{X})} \right\rceil = D. \quad (3.77)$$

Consider the concatenation of ψ_1 and ψ_2 , denoted $\Psi = \Psi_2 \circ \Psi_1$, with $\Psi(\mathcal{X}^q) = \psi_2(\psi_1(\mathcal{X}^q))$.

The total cost is

$$\begin{aligned} T(\Psi) &= \frac{1}{q} \left(\sum_{x^q} P(x^q) W(x^q) \right) \\ &= \frac{1}{q} \left(\sum_{x^q \in A_\epsilon^{(q)}} P(x^q) W(x^q) + \sum_{x^q \in A_\epsilon^{(q)C}} P(x^q) W(x^q) \right) \\ &\leq \frac{1}{q} \left[\sum_{x^q \in A_\epsilon^{(q)}} P(x^q) (\log_2 M/\mu + \max_i C_i) \right. \\ &\quad \left. + \sum_{x^q \in A_\epsilon^{(q)C}} P(x^q) D (\log_2 M/\mu + \max_i C_i) \right] \\ &= \frac{1}{q} [\Pr\{A_\epsilon^{(q)}\} (\log_2 M/\mu + \max_i C_i) \\ &\quad + \Pr\{A_\epsilon^{(q)C}\} D (\log_2 M/\mu + \max_i C_i)] \\ &< \frac{1}{q} [(\log_2 M/\mu + \max_i C_i) \\ &\quad + \epsilon D (\log_2 M/\mu + \max_i C_i)] \\ &= \left(\frac{\log_2 M}{q\mu} + \frac{\max_i C_i}{q} \right) (1 + \epsilon D) \end{aligned} \quad (3.78)$$

where

$$M = 2^{\lceil q(H(\mathbf{X}) + \varepsilon) + 2 \rceil} + \delta. \quad (3.79)$$

Since $\delta \leq |\mathcal{Y}| - 2$, we can bound M by

$$M \leq 2^{q(H(\mathbf{X}) + \varepsilon) + 3} + |\mathcal{Y}| \quad (3.80)$$

and by L'Hôpital's rule, we have

$$\lim_{q \rightarrow \infty} \frac{\log_2(2^{q(H(\mathbf{X}) + \varepsilon) + 3} + |\mathcal{Y}|)}{q} = H(\mathbf{X}) + \varepsilon. \quad (3.81)$$

Therefore, there exists positive integer Q_2 such that, when $q > Q_2$,

$$\frac{\log_2 M}{q} < H(\mathbf{X}) + \varepsilon + \varepsilon'. \quad (3.82)$$

Thus, when $q > \max\{Q_1, Q_2\}$, the total cost of Ψ is upper bounded by

$$T(\Psi) < \left(\frac{H(X) + \varepsilon + \varepsilon'}{\mu} + \frac{\max_i C_i}{q} \right) (1 + \varepsilon D). \quad (3.83)$$

Choose $Q_3 = \left\lceil \frac{8 \max_i C_i}{\gamma} \right\rceil$. When $q > \max\{Q_1, Q_2, Q_3\}$, we have

$$\begin{aligned} T(\Psi) &< \left(\frac{H(\mathbf{X}) + \varepsilon + \varepsilon'}{\mu} + \frac{\max_i C_i}{q} \right) (1 + \varepsilon D) \\ &= \frac{H(\mathbf{X})}{\mu} + \left(\frac{\varepsilon + \varepsilon'}{\mu} + \frac{\max_i C_i}{q} \right) (1 + \varepsilon D) + \varepsilon \frac{H(X)}{\mu} D \\ &\leq \frac{H(\mathbf{X})}{\mu} + \left(\frac{\gamma}{8} + \frac{\gamma}{8} \right) (1 + 1) + \frac{\gamma}{2} = \frac{H(\mathbf{X})}{\mu} + \gamma. \end{aligned} \quad (3.84)$$

As shown in [12, Theorem 3.2.1], for any $\gamma' > 0$, there exists a $Q_4 > 0$ such that when $q > Q_4$,

the average codeword length per input symbol of Ψ_1 satisfies

$$\frac{1}{q}E[L(\Psi_1)] \leq H(\mathbf{X}) + \gamma'. \quad (3.85)$$

The total cost of Ψ_2 for binary uniform i.i.d. source is upper bounded by

$$T'(\Psi_2) \leq \frac{\log_2 M}{\mu \lceil q(H(\mathbf{X}) + \varepsilon) + 2 \rceil} + \frac{\max_i C_i}{\mu \lceil q(H(\mathbf{X}) + \varepsilon) + 2 \rceil}. \quad (3.86)$$

By an argument similar to that used to derive the upper bound on $T(\Psi)$ in (3.84), we conclude that for any $\gamma'' > 0$, there exists a $Q_5 > 0$ such that when $q > Q_5$,

$$T'(\Psi_2) \leq \frac{1}{\mu} + \gamma''. \quad (3.87)$$

To summarize, given any $\gamma, \gamma', \gamma'' > 0$, for sufficiently large q , namely $q > \max\{Q_1, Q_2, Q_3, Q_4, Q_5\}$, we can find a data compression encoder Ψ_1 such that

$$\frac{1}{q}E[L(\Psi_1)] \leq H(\mathbf{X}) + \gamma' \quad (3.88)$$

and a code Ψ_2 for binary uniform i.i.d. source such that

$$T'(\Psi_2) \leq \frac{1}{\mu} + \gamma''. \quad (3.89)$$

The concatenation of Ψ_1 and Ψ_2 will generate a code $\Psi = \Psi_2 \circ \Psi_1$ that has total cost upper bounded by

$$T(\Psi) \leq \frac{H(\mathbf{X})}{\mu} + \gamma. \quad (3.90)$$

This finishes the proof of the separation theorem. $\square$

Remark 3.3.7. Besides modified Varn coding, any fixed-to-variable (or fixed-to-fixed) coding

scheme for a uniform source, such as the constant composition distribution matching codes introduced in [64], can be used to prove the separation theorem, as long as the cost of the codeword sequence is bounded from above and the ratio of the upper bound to input sequence length is asymptotically optimal. $\square$

An example of an optimal type-II shaping scheme that illustrates Theorem 3.3.4 was described by Iwata in [30]. It uses a concatenation of an LZ78 code and a Varn code as outer and inner codes, respectively.

There is also a separation theorem for type-I shaping codes, stating that minimum average cost for a given expansion factor can be achieved by a concatenation of optimal lossless compression with an optimal type-I shaping code for a uniform i.i.d. source and suitable expansion factor. The proof relies on the type-II separation theorem and the equivalence between type-II and type-I shaping codes established in Theorem 3.3.3. It requires an analysis of the behavior of the total cost function in the vicinity of the expansion factor that minimizes total cost.

Theorem 3.3.6. *Given the source $\mathbf{X}$, cost vector C and expansion factor f , the minimum average cost can be achieved by a concatenation of an optimal lossless compression code with a binary optimal type-I shaping code for uniform i.i.d. source and expansion factor*

$$f' = \frac{f}{H(\mathbf{X})}. \quad (3.91)$$

Proof. We know from Theorem 3.3.3 that the problem of designing an optimal type-I shaping code for channel with cost C and expansion f_0 is related to designing an optimal type-II shaping code for channel with cost C' . The minimum total cost for channel with cost C' is $H(\mathbf{X})$ and the expansion factor for an optimal type-II shaping code for channel with cost C' is $f'_0 = f_0$.

We first fix some notation. Given a code ϕ , denote by $T_{C'}(\phi)$ the total cost of this code for channel with cost C' . Denote by $f(\phi)$ its expansion factor and denote by $A_C(\phi)$ and $A_{C'}(\phi)$ the average cost of this code for channel with cost C and channel with cost C' , respectively. The

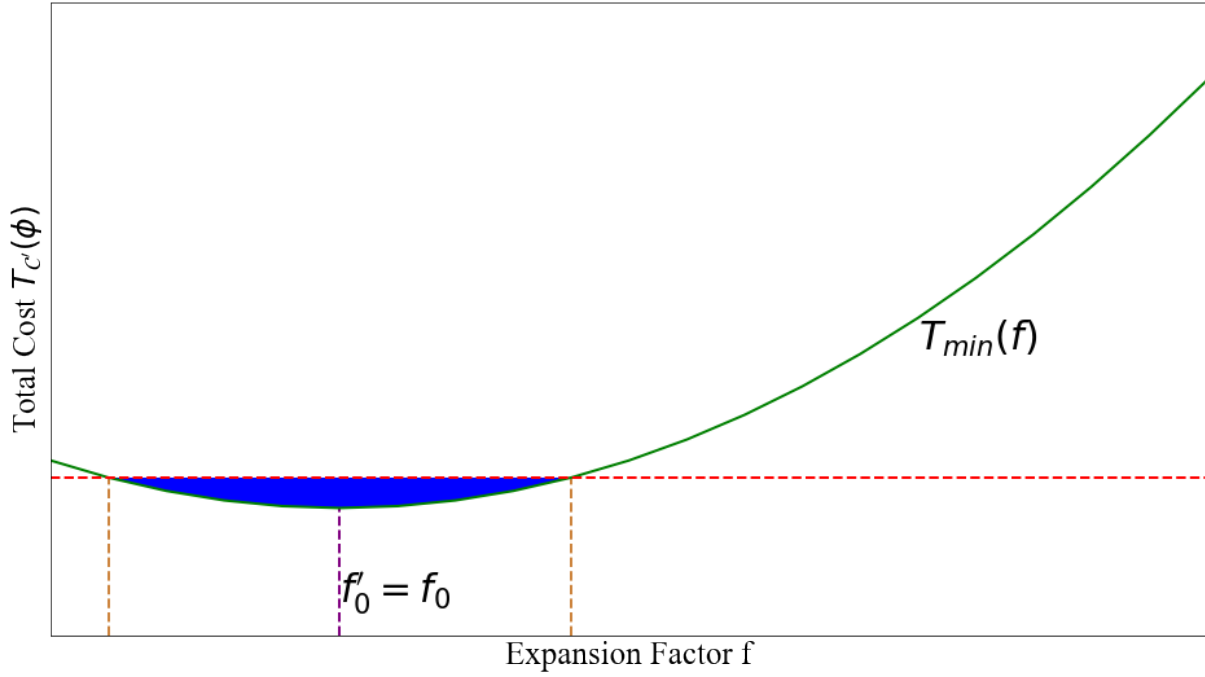


Figure 3.2: Achievable total cost vs. expansion factor

asymptotic symbol occurrence probability of ϕ is $\{P_i\}$ and $\hat{P}_i = \frac{2^{-\mu C_i}}{N}$ is the symbol occurrence probability of the optimal type-I shaping code.

Before we prove the separation theorem for type-I shaping code, we analyze the behavior of $f(\phi)$, $A_C(\phi)$ and $A_{C'}(\phi)$ when $T_{C'}(\phi)$ approaches $H(\mathbf{X})$. The proof of Theorem 3.3.2 determines the achievable total cost on channel with cost C' . By combining equations (3.52) and (3.56), we have

$$\frac{dT_{\min}}{df} = -\frac{1}{\mu} \log_2 N. \quad (3.92)$$

This implies that function $T_{\min}(f)$ is continuous, strictly monotone decreasing on $[\frac{H(\mathbf{X})}{\log_2 |\mathcal{Y}|}, f_0)$ and strictly monotone increasing on (f_0, ∞) . This is shown schematically by the green curve $T_{\min}(f)$ in Fig. 3.2. Thus for any $\zeta > 0$, there exists a γ such that if there exists a code such that $T_{C'}(\phi) < H(\mathbf{X}) + \gamma$, as indicated by the blue area in Fig. 3.2, then $|f(\phi) - f_0| < \zeta$. Such a code can always be found for q sufficiently large, for example, by using the generalized Shannon-Fano construction (see [14]).

Now for any $\eta > 0$, $\zeta > 0$, first choose $\zeta' = \min\{\frac{f_0}{2}, \zeta, \frac{\eta\mu f_0^2}{4H(\mathbf{X})}\}$. Choose γ' such that a type-II shaping code with total cost upper bounded by $H(\mathbf{X}) + \gamma'$ has expansion factor $|f - f_0| < \zeta'$. Choose $\gamma = \min\{\gamma', \frac{\mu\eta f_0}{4}\}$, then $A_{C'}(\phi)$ is bounded by

$$\begin{aligned}
A_{C'}(\phi) &= \sum_i P_i C'_i = \frac{T_{C'}(\phi)}{f} < \frac{H(\mathbf{X}) + \gamma}{f_0 - \zeta'} \\
&= \frac{H(\mathbf{X})}{f_0} + \frac{H(\mathbf{X})}{f_0} \frac{\zeta'}{f_0 - \zeta'} + \frac{\gamma}{f_0 - \zeta'} \\
&\leq \frac{H(\mathbf{X})}{f_0} + \frac{H(\mathbf{X})}{f_0} \frac{2\zeta'}{f_0} + \frac{2\gamma}{f_0} \\
&\leq \frac{H(\mathbf{X})}{f_0} + \frac{\mu\eta}{2} + \frac{\mu\eta}{2} = \frac{H(\mathbf{X})}{f_0} + \mu\eta.
\end{aligned} \tag{3.93}$$

Since the channel cost C'_i is calculated by

$$C'_i = -\log_2 \hat{P}_i = -\log_2 \frac{2^{-\mu C_i}}{N} = \mu C_i + \log_2 N. \tag{3.94}$$

$A_C(\phi)$ is upper bounded by

$$\begin{aligned}
A_C(\phi) &= \sum_i P_i C_i = (\sum_i P_i C'_i - \log_2 N) / \mu \\
&< (\frac{H(\mathbf{X})}{f_0\mu} - \frac{\log_2 N}{\mu}) + \eta.
\end{aligned} \tag{3.95}$$

To summarize, for any $\eta > 0$, $\zeta > 0$, there exists a $\gamma > 0$ such that if there exists a code ϕ such that $T_{C'}(\phi) < H(\mathbf{X}) + \gamma$, this code has expansion factor

$$|f(\phi) - f_0| < \zeta \tag{3.96}$$

and the average cost $A_C(\phi)$ is upper bounded by

$$A_C(\phi) < (\frac{H(\mathbf{X})}{f_0\mu} - \frac{\log_2 N}{\mu}) + \eta. \tag{3.97}$$

Now we present the proof of Theorem 3.3.6. We consider the channel with cost C and equivalent channel with cost C' . Given $f, \eta, \zeta, \eta', \zeta'$, and $\gamma' > 0$, for q sufficiently large, there exists a data compression encoder Ψ_1 such that the average codeword length

$$\frac{1}{q}E[L(\Psi_1)] \leq H(\mathbf{X}) + \gamma' \quad (3.98)$$

and a code Ψ_2 for a binary uniform i.i.d. source and costly channel with cost C' such that

$$T_{C'}(\Psi_2) \leq 1 + \gamma'. \quad (3.99)$$

The expansion factor of an optimal type-II shaping code for binary uniform i.i.d. source and costly channel C' is

$$f' = \frac{f}{H(\mathbf{X})}. \quad (3.100)$$

So when γ'' is small enough, Ψ_2 has expansion factor

$$|f(\Psi_2) - \frac{f}{H(\mathbf{X})}| < \zeta' \quad (3.101)$$

and $A_C(\Psi_2)$ is upper bounded by

$$\begin{aligned} A_C(\Psi_2) &< \left(\frac{1}{\frac{f}{H(\mathbf{X})}\mu} - \frac{\log_2 N}{\mu} \right) + \eta' \\ &= \left(\frac{H(\mathbf{X})}{f\mu} - \frac{\log_2 N}{\mu} \right) + \eta'. \end{aligned} \quad (3.102)$$

The concatenation of Ψ_1 and Ψ_2 will generate a code $\Psi = \Psi_2 \circ \Psi_1$ that has total cost upper bounded by

$$T_{C'}(\Psi) \leq H(\mathbf{X}) + \gamma. \quad (3.103)$$

When γ is small enough, Ψ has expansion factor

$$|f(\Psi) - f| < \zeta \quad (3.104)$$

and $A_C(\Psi)$ is upper bounded by

$$A_C(\Psi) < \left(\frac{H(\mathbf{X})}{f\mu} - \frac{\log_2 N}{\mu} \right) + \eta. \quad (3.105)$$

This completes the proof. $\square$

3.4 Distribution Matching Code Design

Given a target distribution $\{P_i\}$, distribution matching (DM) considers the problem of mapping an i.i.d. sequence of source symbols to an output sequence of symbols that are approximately independent and distributed according to $\{P_i\}$. An optimal DM code must satisfy two conditions: the codeword sequence has symbol occurrence probabilities $\hat{P}_i = P_i$, and the output sequence looks like an i.i.d. sequence. We measure the latter property using the asymptotic normalized KL-divergence defined in Lemma 3.2.5.

It has been shown in Theorems 3.3.1 and 3.3.2 that an optimal shaping code will generate an output sequence such that $\lim_{l \rightarrow \infty} \frac{1}{l} D(Y_1^l || \hat{Y}_1^l) = 0$. Thus the output sequence approximates an i.i.d. sequence with symbol occurrence probability distribution $\{\hat{P}_i\}$. This implies that we can solve the distribution matching problem by designing a corresponding shaping code. In this section, we consider the problem of designing optimal DM codes. We first formulate the problem of *generating an i.i.d. sequence* and then show the connection between DM codes and shaping codes. We then propose a *generalized expansion factor* to measure the performance of a DM code. A comparison of DM code performance measures is also presented.

3.4.1 Problem Formulation

We use the asymptotic normalized Kullback-Leibler divergence [68] to formally define an optimal DM code ϕ for distribution $\{P_i\}$.

Definition 3.4.1. A variable-length mapping $\phi : \mathcal{X}^q \rightarrow \mathcal{Y}^*$ is an optimal DM code for distribution $\{P_i\}$ if

$$\lim_{l \rightarrow \infty} \frac{1}{l} D(Y_1^l || \tilde{Y}_1^l) = 0, \quad (3.106)$$

where $\tilde{\mathbf{Y}}$ is an i.i.d. process with distribution $\{P_i\}$. □

By combining Theorem 3.2.1 and Lemma 3.2.5, we can prove the following theorem.

Theorem 3.4.1. *The expansion factor of a mapping satisfies the lower bound*

$$f = \frac{H(\mathbf{X})}{H(\mathbf{Y})} \geq \frac{H(\mathbf{X})}{H(\hat{\mathbf{Y}})} \quad (3.107)$$

with equality if and only if $\lim_{l \rightarrow \infty} \frac{1}{l} D(Y_1^l || \hat{Y}_1^l) = H(\hat{\mathbf{Y}}) - H(\mathbf{Y}) = 0$. When $f = \frac{H(\mathbf{X})}{H(\hat{\mathbf{Y}})}$, this code is an optimal DM code for distribution $\{P_i\}$. □

Remark 3.4.1. Assuming this mapping is an optimal compression, the compression ratio g is

$$g = \frac{H(\mathbf{X})}{\log_2 |\mathcal{Y}|}. \quad (3.108)$$

By Theorem 3.2.1 and Lemma 3.2.5, we have

$$H(\hat{\mathbf{Y}}) \geq H(\mathbf{Y}) = \frac{H(\mathbf{X})}{g} = \log_2 |\mathcal{Y}|. \quad (3.109)$$

Since $H(\hat{\mathbf{Y}}) \leq \log_2 |\mathcal{Y}|$, we know that $H(\hat{\mathbf{Y}}) = H(\mathbf{Y}) = \log_2 |\mathcal{Y}|$ and

$$\lim_{l \rightarrow \infty} \frac{1}{l} D(Y_1^l || \hat{Y}_1^l) = 0. \quad (3.110)$$

This implies the codeword sequence looks i.i.d. and has probability of occurrence

$$\hat{P}_i = \frac{1}{|\mathcal{Y}|} \quad \text{for all } i. \quad (3.111)$$

This proves the well-known fact that the output of an optimal compression approximates a uniform i.i.d. sequence [79],[25],[26]. $\square$

Let $\tilde{\mathbf{Y}}$ be the i.i.d. process with distribution $\{P_i\}$. As in the derivation of (3.31) in Lemma 3.2.5, we find

$$\begin{aligned} & \lim_{l \rightarrow \infty} \frac{1}{l} D(Y_1^l \| \tilde{Y}_1^l) \\ &= - \lim_{l \rightarrow \infty} \frac{1}{l} H(Y_1^l) - \sum_i \log_2 P_i \lim_{l \rightarrow \infty} \sum_{y_1^l \in \mathcal{Y}^l} \frac{Q(y_1^l) N_i(y_1^l)}{l} \\ &= -H(\mathbf{Y}) - \sum_i \hat{P}_i \log_2 P_i = - \sum_i \hat{P}_i \log_2 P_i - \frac{H(\mathbf{X})}{f} \\ &= \frac{-f \sum_i \hat{P}_i \log_2 P_i - H(\mathbf{X})}{f}. \end{aligned} \quad (3.112)$$

From Theorem 3.3.2, we know that for a channel with cost $\{C_i = -\log_2 P_i\}$, the total cost $-f \sum_i \hat{P}_i \log_2 P_i$ is lower bounded by $H(\mathbf{X})$. The shaping code that achieves this lower bound has the following two properties:

- The probability of occurrence of symbol β_i satisfies $\hat{P}_i = P_i$ for all β_i ,
- The asymptotic normalized KL-divergence between $\mathbf{Y}$ and $\hat{\mathbf{Y}}$ satisfies

$$\lim_{l \rightarrow \infty} \frac{1}{l} D(Y_1^l \| \hat{Y}_1^l) = 0. \quad (3.113)$$

This implies that this code generates a sequence that approximates an i.i.d. sequence with

distribution $\{P_i\}$. This analysis also implies that the expansion factor of an optimal DM code is

$$f_{\text{opt}} = \frac{H(\mathbf{X})}{-\sum_i \hat{P}_i \log_2 \hat{P}_i} = \frac{H(\mathbf{X})}{-\sum_i P_i \log_2 P_i}. \quad (3.114)$$

We summarize in the following theorem the relationship between optimal shaping codes and optimal DM codes, extending the result in [26] by explicitly showing the optimal expansion factor.

Theorem 3.4.2. *The optimal type-II shaping code with cost vector C , or the equivalent type-I shaping code from Theorem 3.3.3, is an optimal DM code for distribution $\{P_i\}$ if*

$$C_i = -\log_2 P_i \quad (3.115)$$

for every symbol β_i , in the sense that

$$\lim_{l \rightarrow \infty} \frac{1}{l} D(Y_1^l || \tilde{Y}_1^l) = 0. \quad (3.116)$$

The expansion factor of this optimal DM code is

$$f_{\text{opt}} = \frac{H(\mathbf{X})}{-\sum_i P_i \log_2 P_i}. \quad (3.117)$$

□

Remark 3.4.2. Shell mapping was used in [65] to design fixed-length DM codes with uniformly distributed input bits. The shell mapper that minimizes informational divergence (introduced later in Section 3.6.1) uses the “self-information” weight function $C_i = -\log_2 P_i$ and the optimal expansion factor is determined by a search. Theorem 3.4.2 considers a more general variable-length DM code with arbitrary i.i.d. source and characterizes the optimal expansion factor. Codes minimizing informational divergence are discussed further in Section 3.6.1. □

3.5 Generalized Expansion Factor

The relationship between optimal shaping codes and optimal DM codes was established above. The total cost of the shaping code suggests an alternative performance measure for a DM code which will be useful when analyzing the optimality of a shaping-based DM code construction and in proving a separation theorem for DM codes. Specifically, we define the *generalized expansion factor* (GEF) of a prefix-free variable-length code as follows.

Definition 3.5.1. Given a prefix-free variable-length code $\phi : \mathcal{X}^q \rightarrow \mathcal{Y}^*$ and a set of positive real numbers $\{P_1, P_2, \dots, P_v\}$ such that $\sum P_i = 1$, the **generalized expansion factor** of this code is defined as

$$F(\phi, P_1, \dots, P_v) = -f \frac{\sum_i \hat{P}_i \log_2 P_i}{\log_2 |\mathcal{Y}|} \quad (3.118)$$

where f is the code expansion factor and $\{\hat{P}_i\}$ is the asymptotic symbol occurrence probability distribution. □

For simplicity, we sometimes use F to represent $F(\phi, P_1, \dots, P_v)$. The following theorem shows that F can be used to evaluate an optimal DM code.

Theorem 3.5.1. Given a prefix-free variable-length code $\phi : \mathcal{X}^q \rightarrow \mathcal{Y}^*$ and a set of positive real numbers $\{P_1, P_2, \dots, P_v\}$ such that $\sum P_i = 1$, the generalized expansion factor of this mapping is lower bounded by

$$F(\phi, P_1, \dots, P_v) \geq \frac{H(\mathbf{X})}{\log_2 |\mathcal{Y}|}. \quad (3.119)$$

If $F = \frac{H(\mathbf{X})}{\log_2 |\mathcal{Y}|}$, this mapping is an optimal DM code for the target distribution $\{P_i\}$, in the sense that

$$\lim_{l \rightarrow \infty} \frac{1}{l} D(Y_1^l || \tilde{Y}_1^l) = 0. \quad (3.120)$$

□

Proof. Assume symbol β_i in the codeword sequence has cost $C_i = -\log_2 P_i$. The total cost of

this mapping is

$$T(\phi) = f \sum_i \hat{P}_i C_i = -f \sum_i \hat{P}_i \log_2 P_i. \quad (3.121)$$

Comparing equations (3.118) and (3.121) we have

$$F(\phi, P_1, \dots, P_v) = \frac{T(\phi)}{\log_2 |\mathcal{Y}|}. \quad (3.122)$$

This indicates that the GEF of a DM code is equivalent to its total cost when applying it to a costly channel with cost $C_i = -\log_2 P_i$. From Theorem 3.3.2, we know the total cost of a prefix-free mapping satisfies the lower bound

$$T(\phi) \geq \frac{H(\mathbf{X})}{\mu} \quad (3.123)$$

where μ is a constant such that $\sum 2^{-\mu C_i} = 1$. Since $C_i = -\log_2 P_i$, it is easy to check that $\mu = 1$ and

$$F(\phi, P_1, \dots, P_v) = \frac{T(\phi)}{\log_2 |\mathcal{Y}|} \geq \frac{H(\mathbf{X})}{\log_2 |\mathcal{Y}|}. \quad (3.124)$$

When the minimum GEF is achieved, this code is also an optimal type-II shaping code with $C_i = -\log_2 P_i$. Theorem 3.4.2 then implies that this code is an optimal DM code for the target distribution $\{P_i\}$, in the sense that

$$\lim_{l \rightarrow \infty} \frac{1}{l} D(Y_1^l || \tilde{Y}_1^l) = 0. \quad (3.125)$$

□

Remark 3.5.1. As shown in Remark 3.3.6, for a uniform i.i.d. source and a cost vector $\mathcal{C}$, a Varn code $\phi_K : \mathcal{X}^{\log_{|\mathcal{X}|} K} \rightarrow \mathcal{Y}^*$ is an asymptotically optimal type-II shaping code. If the costs are given by $C_i = -\log_2 P_i$, where $\sum_i P_i = 1$, the total cost is bounded by

$$\log_2 |\mathcal{X}| \leq T(\phi_K) \leq \log_2 |\mathcal{X}| + \frac{\max_i \{C_i\}}{\log_{|\mathcal{X}|} K}. \quad (3.126)$$

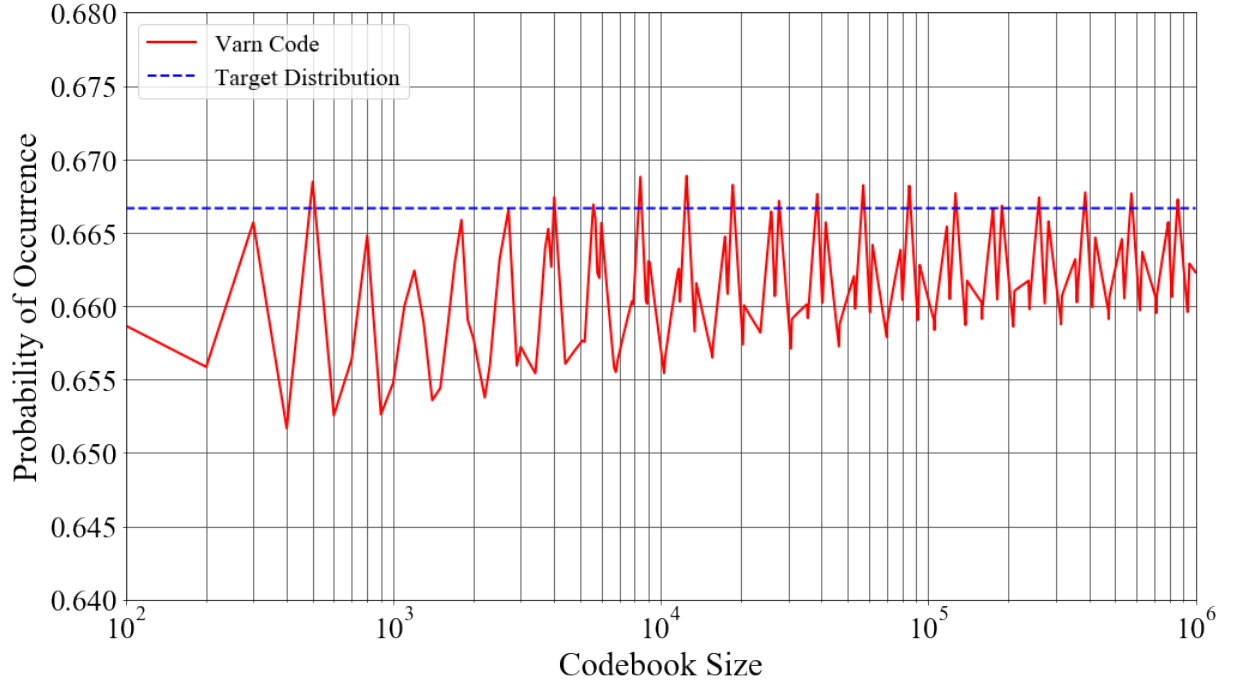


Figure 3.3: Probability of occurrence $\hat{P}_0$ of a Varn code for the target distribution $\{2/3, 1/3\}$.

Equation (3.122) implies that for the target distribution $\{P_i\}$, Varn codes minimize GEF for a specified codebook size K . Thus, a Varn code can be regarded as a DM code with generalized expansion factor bounded by

$$\frac{\log_2 |\mathcal{X}|}{\log_2 |\mathcal{Y}|} \leq F \leq \frac{\log_2 |\mathcal{X}|}{\log_2 |\mathcal{Y}|} \left(1 + \frac{\max_i \{-\log_2 P_i\}}{\log_2 K}\right). \quad (3.127)$$

Therefore, we have

$$\lim_{K \rightarrow \infty} F(\phi_K, P_1, \dots, P_v) = \frac{\log_2 |\mathcal{X}|}{\log_2 |\mathcal{Y}|} \quad (3.128)$$

which implies that Varn codes are asymptotically optimal DM codes. Fig. 3.3 and Fig. 3.4 show the probability of occurrence and generalized expansion factor of binary Varn codes (i.e., with $\mathcal{X} = \mathcal{Y} = \{0, 1\}$) for a target distribution $P_0 = 2/3, P_1 = 1/3$. As the codebook size K increases, we see that the probability of occurrence $\hat{P}_0$ approaches the target distribution value $P_0 = 2/3$ and the generalized expansion factor approaches the theoretical lower bound 1. $\square$

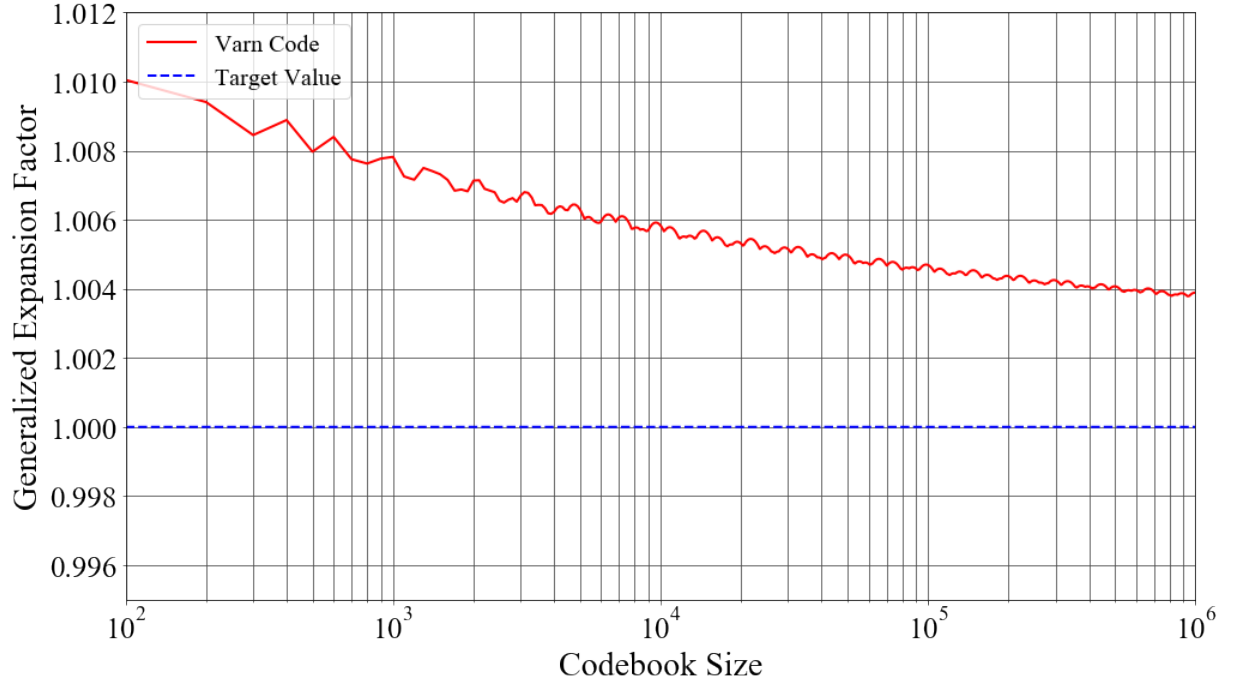


Figure 3.4: Generalized expansion factor of a Varn code for the target distribution $\{2/3, 1/3\}$.

The separation theorem for shaping codes in Theorem 3.3.4 now extends naturally to DM codes.

Theorem 3.5.2. *An optimal DM code can be constructed by a concatenation of optimal lossless compression with an optimal DM code for a uniform i.i.d. source, in the sense that the minimum generalized expansion factor can be achieved by such a concatenation.* $\square$

Remark 3.5.2. When $P_1 = P_2 = \dots = P_v = \frac{1}{|\mathcal{Y}|}$, the generalized expansion factor reduces to

$$F = f \frac{\sum_i \hat{P}_i \log_2 |\mathcal{Y}|}{\log_2 |\mathcal{Y}|} = f. \quad (3.129)$$

This provides the motivation for designating F by this name. $\square$

Remark 3.5.3. We use an example to illustrate the difference between the generalized expansion factor and the normalized conditional divergence introduced in [26] and [25] when the encoder has finite length. Given a ternary source with alphabet $\mathcal{X} = \{\alpha_1, \alpha_2, \alpha_3\}$ and probability distribution

$\{\frac{1}{2}, \frac{1}{4}, \frac{1}{4}\}$, consider two codes defined by the mappings

$$\begin{aligned}\Phi_1 : \{\alpha_1 \rightarrow 0, \alpha_2 \rightarrow 10, \alpha_3 \rightarrow 11\}, \\ \Phi_2 : \{\alpha_1 \rightarrow 00, \alpha_2 \rightarrow 10, \alpha_3 \rightarrow 11\}.\end{aligned}\tag{3.130}$$

Their generalized expansion factors for target distribution $\{1/2, 1/2\}$ are

$$F_1 = \frac{3}{2} < F_2 = 2.\tag{3.131}$$

This suggests that Φ_1 is a better approximation of an optimal DM code for target distribution $\{\frac{1}{2}, \frac{1}{2}\}$ (in fact, Φ_1 is an optimal DM code). The normalized conditional divergences are

$$\begin{aligned}D(\Phi_1(X)||V|I) = \\ \frac{1}{2}(\log_2 \frac{1}{1/2}) + \frac{1}{2}(\frac{1}{2}\log_2 \frac{1/2}{1/4} + \frac{1}{2}\log_2 \frac{1/2}{1/4}) = 1\end{aligned}\tag{3.132}$$

$$\begin{aligned}D(\Phi_2(X)||V|I) \\ = \frac{1}{2}\log_2 \frac{1/2}{1/4} + \frac{1}{4}\log_2 \frac{1/4}{1/4} + \frac{1}{4}\log_2 \frac{1/4}{1/4} = \frac{1}{2}.\end{aligned}\tag{3.133}$$

We find that $D(\Phi_1(X)||V|I) > D(\Phi_2(X)||V|I)$, which suggests the opposite conclusion that Φ_2 would be a better approximation of the optimal DM code. $\square$

3.6 Comparison of DM Performance Measures

In this section, we use a shaping code perspective to study DM codes whose performance is measured using informational divergence and normalized informational divergence.

3.6.1 Generalized Expansion Factor and Informational Divergence

In this subsection we study the relationship between the generalized expansion factor and the informational divergence introduced in [3], which is also used as a performance measure for DM codes.

Consider a variable-length code $\phi : \mathcal{X}^{\log_{|\mathcal{X}|} K} \rightarrow \mathcal{Y}^*$ with codebook size K . We use $\mathcal{L}$ to denote the set of all codewords generated by this mapping. The leaf probability, or the probability of a codeword y_1^l , is defined as

$$P^{\mathcal{L}}(y_1^l) = P(y_1)P(y_2) \dots P(y_l) = \prod P_i^{N_i(y_1^l)}. \quad (3.134)$$

This is also the probability of sequence y_1^l generated by an i.i.d. source with distribution $\{P_i\}$. The true probability of codeword y_1^l is the probability of the corresponding source sequence $\phi^{-1}(y_1^l)$. The *informational divergence* (I-divergence) between these two distributions is defined as

$$I = \sum_{y_1^l \in \mathcal{L}} P(\phi^{-1}(y_1^l)) \log_2 \frac{P(\phi^{-1}(y_1^l))}{P^{\mathcal{L}}(y_1^l)}. \quad (3.135)$$

Now we use the same code for type-II shaping. We set the cost of each symbol to be $C_i = -\log_2 P_i$. The cost of codeword y_1^l is

$$\begin{aligned} W(y_1^l) &= \sum C_i N_i(y_1^l) = -\sum \log_2 P_i^{N_i(y_1^l)} \\ &= -\log_2 \prod P_i^{N_i(y_1^l)} = -\log_2 P^{\mathcal{L}}(y_1^l) \end{aligned} \quad (3.136)$$

and the total cost of this shaping code, or equivalently the GEF, is

$$\begin{aligned}
F(\phi, P_1, \dots, P_v) &= \frac{T(\phi)}{\log_2 |\mathcal{Y}|} \\
&= \frac{1}{\log_2 |\mathcal{X}| K \log_2 |\mathcal{Y}|} \sum_{y_1^l \in \mathcal{L}} P(\phi^{-1}(y_1^l)) W(y_1^l) \\
&= -\frac{\log_2 |\mathcal{X}|}{\log_2 K \log_2 |\mathcal{Y}|} \sum_{y_1^l \in \mathcal{L}} P(\phi^{-1}(y_1^l)) \log_2 P^{\mathcal{L}}(y_1^l).
\end{aligned} \tag{3.137}$$

The I-divergence of this code can then be expressed in terms of its GEF, namely

$$\begin{aligned}
I &= \sum_{y_1^l \in \mathcal{L}} P(\phi^{-1}(y_1^l)) \log_2 \frac{P(\phi^{-1}(y_1^l))}{P^{\mathcal{L}}(y_1^l)} \\
&= \sum_{y_1^l \in \mathcal{L}} P(\phi^{-1}(y_1^l)) \log_2 P(\phi^{-1}(y_1^l)) \\
&\quad - \sum_{y_1^l \in \mathcal{L}} P(\phi^{-1}(y_1^l)) \log_2 P^{\mathcal{L}}(y_1^l) \\
&= F \frac{\log_2 K \log_2 |\mathcal{Y}|}{\log_2 |\mathcal{X}|} - H(\mathbf{X}^{\log_2 |\mathcal{X}| K}) \\
&= (F - \frac{H(\mathbf{X})}{\log_2 |\mathcal{Y}|}) \frac{\log_2 K \log_2 |\mathcal{Y}|}{\log_2 |\mathcal{X}|}.
\end{aligned} \tag{3.138}$$

Since $\log_2 K$ is a constant, minimizing I is equivalent to minimizing F . This equation shows the relationship between I-divergence and GEF, and also highlights the duality between costly channel coding and DM coding. As shown in Remark 3.5.1, Varn codes minimize GEF for a uniform i.i.d. source. Therefore we can conclude the following optimality theorem for Varn codes.

Theorem 3.6.1. *Let $\{P_i\}$ be a target distribution. A code $\phi : \mathcal{X}^{\log_2 |\mathcal{X}| K} \rightarrow \mathcal{Y}^*$ for a uniform i.i.d. source that minimizes I-divergence is given by a Varn code designed for costs $C_i = -\log_2 P_i$. $\square$*

Remark 3.6.1. In [61], Savari showed that Varn codes and reverse Tunstall codes are identical when finding exhaustive prefix-free codes (i.e., when $(K-1)/(|\mathcal{Y}|-1)$ is an integer). Specifically,

a Tunstall code designed to compress distribution $\{P_i\}$ and a Varn code designed for costly channel $\{C_i = -\log_2 P_i\}$ generate identical code trees. Therefore a reverse Tunstall code minimizes the I-divergence when the target distribution is binary (i.e., when $|\mathcal{Y}| = 2$). This was also proved in [3, Proposition 1] using a different method.

However, for non-exhaustive codes this equivalence does not exist, and it remains unknown whether a reverse Tunstall code minimizes I-divergence when the target distribution is non-binary (i.e., when $|\mathcal{Y}| > 2$). Therefore Theorem 3.6.1 can be viewed as a generalization of [3, Proposition 1]. $\square$

3.6.2 Type-I Shaping Problem and Normalized I-Divergence

Another measure for DM codes used in [3] is normalized I-divergence. In this subsection, we study its properties using the perspective of the type-I shaping problem. Normalized I-divergence is defined as

$$\mathcal{J} = \frac{I}{E(L)}. \quad (3.139)$$

Using (3.138), we rewrite this as

$$\begin{aligned} \mathcal{J} &= \left(F - \frac{H(\mathbf{X})}{\log_2 |\mathcal{Y}|}\right) \frac{\log_2 K \log_2 |\mathcal{Y}|}{E(L) \log_2 |\mathcal{X}|} \\ &= \left(-f \frac{\sum_i \hat{P}_i \log_2 P_i}{\log_2 |\mathcal{Y}|} - \frac{H(\mathbf{X})}{\log_2 |\mathcal{Y}|}\right) \frac{\log_2 |\mathcal{Y}|}{f} \\ &= \sum_i \hat{P}_i C_i - \frac{H(\mathbf{X})}{f} \end{aligned} \quad (3.140)$$

where $C_i = -\log_2 P_i$. From equations (3.112) and (3.140) we see that asymptotic normalized KL-divergence and normalized I-divergence are identical for i.i.d. distribution matching.

We divide the problem of finding the minimum $\mathcal{J}$ into two parts. First we fix the expansion factor f and find the minimum achievable $\mathcal{J}$, denoted by $\mathcal{J}_{\min}(f)$. Then we find the optimal f to minimize $\mathcal{J}_{\min}(f)$. The result is found by noting the similarity to the type-I shaping

problem and invoking Theorem 3.3.1.

Theorem 3.6.2. *Let ϕ be a prefix-free variable-length mapping with expansion factor f . Let $\{P_i\}$ be the target distribution and set $C_i = -\log_2 P_i$. The minimum normalized I-divergence $\mathcal{J}_{\min}(f)$ with fixed f is*

$$\mathcal{J}_{\min}(f) = \sum_i \hat{P}_i C_i - \frac{H(\mathbf{X})}{f}, \quad (3.141)$$

where $\hat{P}_i = \frac{2^{-\mu C_i}}{\sum_i 2^{-\mu C_i}}$ and $H(\hat{Y}) = -\sum_i \hat{P}_i \log_2 \hat{P}_i = H(\mathbf{X})/f$.

Proof. We must solve the following optimization problem, which is closely related to the type-I shaping problem.

$$\begin{aligned} & \underset{\hat{P}_i}{\text{minimize}} && \sum_i \hat{P}_i C_i - \frac{H(\mathbf{X})}{f} \\ & \text{subject to} && H(\hat{Y}) \geq \frac{H(\mathbf{X})}{f} \\ & && \sum_i \hat{P}_i = 1. \end{aligned} \quad (3.142)$$

From Theorem 3.3.1, we immediately have

$$\mathcal{J}_{\min}(f) = \sum_i \hat{P}_i C_i - \frac{H(\mathbf{X})}{f} = \sum_i \hat{P}_i C_i - H(\hat{Y}), \quad (3.143)$$

where $\hat{P}_i = \frac{2^{-\mu C_i}}{\sum_j 2^{-\mu C_j}}$ and $H(\hat{Y}) = -\sum_i \hat{P}_i \log_2 \hat{P}_i = H(\mathbf{X})/f$. □

The next proposition determines the derivative of $\mathcal{J}_{\min}(f)$ and finds the optimal expansion factor, f_{opt} , that minimizes $\mathcal{J}_{\min}(f)$.

Proposition 3.6.3. *The first derivative of $\mathcal{J}_{\min}(f)$ is*

$$\frac{d\mathcal{J}_{\min}}{df} = \frac{H(\mathbf{X})}{f^2} \frac{\mu - 1}{\mu} \quad \mu > 0. \quad (3.144)$$

Let $f_{\text{opt}} = -H(\mathbf{X})/\sum_i P_i \log_2 P_i$. Then $\mathcal{J}_{\min}(f)$ is continuous, strictly monotone decreasing on $[\frac{H(\mathbf{X})}{\log_2 |\mathcal{Y}|}, f_{\text{opt}})$ (or, for $\mu \in [0, 1)$) and continuous, strictly monotone increasing on $(f_{\text{opt}}, +\infty)$ (or,

for $\mu \in (1, \infty)$). When $f = f_{\text{opt}}$, $\mathcal{J}_{\min}(f_{\text{opt}}) = 0$.

Proof. We have studied the behavior of minimum total cost with fixed f in Theorem 3.3.2 and 3.3.6. Here we use the same technique to study $\mathcal{J}_{\min}(f)$. Note that $\mathcal{J}_{\min}(f)$ is a function of μ . The derivative $df/d\mu$ is already given in equation (3.52), and it is easy to check that

$$\frac{d\mathcal{J}_{\min}}{d\mu} = \frac{(\mu - 1) \ln 2 \sum_{i < j} 2^{-\mu(C_i + C_j)} (C_i - C_j)^2}{N^2}. \quad (3.145)$$

Applying the chain rule along with (3.145) and (3.52), we have

$$\frac{d\mathcal{J}_{\min}}{df} = \frac{H(\mathbf{X})}{f^2} \frac{\mu - 1}{\mu}. \quad (3.146)$$

Let $f_{\text{opt}} = \frac{H(\mathbf{X})}{-\sum_i P_i \log_2 P_i}$ (or, equivalently, let $\mu = 1$). Equations (3.145) and (3.146) imply that $\mathcal{J}_{\min}(f)$ is continuous, strictly monotone decreasing on $[\frac{H(\mathbf{X})}{\log_2 |\mathcal{Y}|}, f_{\text{opt}})$ (or, for $\mu \in [0, 1)$) and strictly monotone increasing on $(f_{\text{opt}}, +\infty)$ (or, for $\mu \in (1, \infty)$). The minimum of $\mathcal{J}_{\min}(f)$ is achieved when $f = f_{\text{opt}}$. We have

$$\begin{aligned} \mathcal{J}_{\min}(f_{\text{opt}}) &= \sum_i \hat{P}_i C_i - \frac{H(\mathbf{X})}{f_{\text{opt}}} \\ &= \sum_i \frac{2^{-C_i}}{\sum_j 2^{-C_j}} C_i + \sum_i P_i \log_2 P_i \\ &= -\sum_i \frac{P_i}{\sum_j P_j} \log_2 P_i + \sum_i P_i \log_2 P_i = 0. \end{aligned} \quad (3.147)$$

This completes the proof. □

Remark 3.6.2. In [5, Sec 5.1], the author studied the minimum KL-divergence between a pmf $\{\hat{P}_i\}$ and the target distribution $\{P_i\}$, where each $\hat{P}_i$ is associated with a cost w_i and the average cost of the pmf is upper bounded by C . The analysis is similar to the analysis in Proposition 3.6.3

if we specialize to the case where $w_i = -\log_2 P_i = C_i$. The KL-divergence is

$$\begin{aligned}
D &= D(\hat{P}_i || P_i) = \sum_i \hat{P}_i \log_2 \frac{\hat{P}_i}{P_i} \\
&= -\sum_i \hat{P}_i \log_2 P_i + \sum_i \hat{P}_i \log_2 \hat{P}_i \\
&= \sum_i \hat{P}_i C_i - H(\hat{P}).
\end{aligned} \tag{3.148}$$

By combining equation (3.140) with Lemma 3.2.5, we have

$$\begin{aligned}
\mathcal{J} &= \sum_i \hat{P}_i C_i - \frac{H(\mathbf{X})}{f} = \sum_i \hat{P}_i C_i - H(\mathbf{Y}) \\
&\geq \sum_i \hat{P}_i C_i - H(\hat{P}) = D,
\end{aligned} \tag{3.149}$$

with equality if and only if the output process generated by ϕ approximates an i.i.d. process (Remark 3.2.4).

The minimum KL-divergence with average cost upper bounded by a specified average cost C , denoted by $D(C)$, was also studied in [5, Sec 5.1]. The pmf that achieves $D(C)$ is

$$\hat{P}_i = \frac{2^{-\mu C_i}}{\sum_j 2^{-\mu C_j}}, \quad \sum_i \hat{P}_i C_i = C, \tag{3.150}$$

when $C \leq \sum P_i C_i$, or $\mu \geq 1$. When $C > \sum P_i C_i$, by setting $\mu = 1$, we have $\hat{P}_i = P_i$, $\sum_i \hat{P}_i C_i < C$, and

$$D(C) = \sum_i \hat{P}_i C_i - H(\hat{P}) = 0. \tag{3.151}$$

By comparing equations (3.141) and (3.150), we conclude that

$$\mathcal{J}_{\min}(f) = D(C), \tag{3.152}$$

where

$$C = \sum_i \hat{P}_i C_i \quad f = \frac{H(\mathbf{X})}{-\sum_i \hat{P}_i \log_2 \hat{P}_i} \quad \hat{P}_i = \frac{2^{-\mu C_i}}{\sum_j 2^{-\mu C_j}}, \quad (3.153)$$

when $\mu \geq 1$, or equivalently when $C \leq \sum P_i C_i$ and $f \geq f_{\text{opt}}$. $\square$

The derivative $dD(C)/dC$ was found in [5, Sec. 5.1]. Using the chain rule, we know that

$$\frac{dD(C)}{dC} = \begin{cases} \frac{d\mathcal{J}_{\min}}{df} \frac{df}{dC} & \text{when } C \leq \sum P_i C_i \ (\mu \geq 1), \\ 0 & \text{when } C > \sum P_i C_i. \end{cases} \quad (3.154)$$

From (3.42) we have

$$\frac{dh}{dC} = \frac{d\frac{H(\mathbf{X})}{f}}{dC} = \mu \quad \Rightarrow \quad \frac{df}{dC} = -\frac{\mu f^2}{H(\mathbf{X})}. \quad (3.155)$$

By combining this with (3.144), we have

$$\frac{dD(C)}{dC} = \begin{cases} 1 - \mu & \text{when } C \leq \sum P_i C_i \ (\mu \geq 1), \\ 0 & \text{when } C > \sum P_i C_i. \end{cases} \quad (3.156)$$

Therefore Proposition 3.6.3 allows us to recover the derivative of $D(C)$. $\square$

Remark 3.6.3. In [6, Sec. V], a bound on the rate of a prefix-free variable-length DM code in the vicinity of $\mathcal{J} = 0$ was given. Proposition 3.6.3 gives an explicit relationship between $\mathcal{J}$ and the code rate over a wider range of rates. $\square$

Remark 3.6.4. In [68, Section 6.2.2], Soriaga considered the case where system requirements dictate that the expansion factor of the DM code encoder can not exceed f_0 , where $f_0 < f_{\text{opt}}$. In such a case, the code cannot be an optimal DM code for target distribution $\{P_i\}$. We may try to approximate an optimal DM code by designing a code with $f \leq f_0$ that minimizes the asymptotic normalized KL-divergence, $\lim_{l \rightarrow \infty} \frac{1}{l} D(Y_1^l || \tilde{Y}_1^l)$. We denote by $\mathcal{D}(f_0)$ the minimum possible value of this divergence. The relationship between $\mathcal{D}(f_0)$ and f_0 for a finite-order Markov target distribution was given in [68] and the result is also applicable to the i.i.d. case

considered here. Since the asymptotic normalized KL-divergence for a fixed f is lower bounded by $\mathcal{J}_{\min}(f)$ (Theorem 3.6.2) and $\mathcal{J}_{\min}(f)$ is strictly monotone decreasing when $f \leq f_0 < f_{\text{opt}}$ (Proposition 3.6.3), we have

$$\begin{aligned}
\mathcal{D}(f_0) &= \mathcal{J}_{\min}(f_0) = \sum_i \hat{P}_i C_i - \frac{H(\mathbf{X})}{f_0} \\
&= \sum_i \hat{P}_i \frac{(-\log_2 \hat{P}_i - \log_2 N)}{\mu} - \frac{H(\mathbf{X})}{f_0} \\
&= \frac{-\sum_i \hat{P}_i \log_2 \hat{P}_i}{\mu} - \sum_i \hat{P}_i \frac{\log_2 N}{\mu} - \frac{H(\mathbf{X})}{f_0} \\
&= \frac{H(\mathbf{X})}{\mu f_0} - \frac{\log_2 N}{\mu} - \frac{H(\mathbf{X})}{f_0},
\end{aligned} \tag{3.157}$$

where μ and N are constants such that $N = \sum_i 2^{-\mu C_i}$ and $\sum_i -\hat{P}_i \log \hat{P}_i = H(\mathbf{X})/f_0$, with $\hat{P}_i = \frac{1}{N} 2^{-\mu C_i}$. Because the code that achieves this lower bound with expansion factor f_0 is an optimal type-I shaping code, based on the equivalence theorem we can extend the result in [68, Section 6.2.2] by concluding that this code is an optimal DM code for target distribution $\{\hat{P}_i\}$. $\square$

Remark 3.6.5. In Theorem 3.6.2, we have shown that when $\mathcal{J} \rightarrow 0$, then $f \rightarrow f_{\text{opt}}$. This implies that the GEF of the code satisfies

$$F = \frac{f \mathcal{J} + H(\mathbf{X})}{\log_2 |\mathcal{Y}|} \rightarrow \frac{H(\mathbf{X})}{\log_2 |\mathcal{Y}|}. \tag{3.158}$$

Similarly, as shown in Appendix 3.3.6, when $F \rightarrow H(\mathbf{X})/\log_2 |\mathcal{Y}|$ (or, equivalently, when total cost $T \rightarrow H(\mathbf{X})$), then $f \rightarrow f_{\text{opt}}$ and

$$\mathcal{J} = \frac{F - \frac{H(\mathbf{X})}{\log_2 |\mathcal{Y}|}}{f} \rightarrow 0. \tag{3.159}$$

In view of the equivalence between asymptotic normalized KL-divergence and $\mathcal{J}$, these observations extend Theorem 3.5.1 by providing bounds on asymptotic normalized KL-divergence in the

vicinity of $F = H(\mathbf{X})/\log_2 |\mathcal{Y}|$. □

3.7 Experimental Results

3.7.1 Optimal Data Shaping Code for MLC Flash Memory

We evaluated the performance of shaping codes on a multilevel-cell (MLC) NAND flash memory. In MLC flash, the cells are arranged in a rectangular array (also called a *block*) and each row of cells is called a *wordline*. The cells can be programmed to four different voltage levels, denoted $\{0, 1, 2, 3\}$, so each cell can store two bits of information. It was shown in [44], [43] that MLC flash memory can be modeled as a costly channel with alphabet $\{0, 1, 2, 3\}$, where the cost of the erase level 0 can be taken to be $C_0 = 0$. Using the methodology described in [44], the cost vector for the memory was found empirically to be

$$\mathcal{C} = [0, 0.58, 0.87, 1.29]. \quad (3.160)$$

From Theorem 3.3.2, we know that the total cost is a decreasing function of the expansion factor. To assess the performance of optimal shaping, and to permit a comparison to the direct-shaping code in [44], [43], we applied a rate-1, type-I shaping code to the ASCII representation of the English-language text of *The Count of Monte Cristo*. The “optimal” shaping scheme was designed according to the principles suggested by the equivalence theorem and separation theorem. We first compressed the file using the LZ77 algorithm. The observed compression rate was $g = 1/2.740$. We then used Theorem 3.3.1 to compute the target symbol occurrence probabilities of a shaping code that minimizes average cost for a uniform i.i.d. source, a cost vector $\mathcal{C}$, and the expansion factor $f' = f/g = 2.740$. The resulting symbol occurrence probability distribution was given by

$$\hat{P} = [0.8606, 0.0989, 0.0335, 0.0070]. \quad (3.161)$$

Using Theorem 3.3.3, we computed the costs for the equivalent code that minimizes total cost, yielding the cost vector

$$\mathcal{C}' = [0.2167, 3.3378, 4.8983, 7.1585]. \quad (3.162)$$

We constructed a Varn code with codebook size $K = 256$ based on the cost vector $\mathcal{C}'$. This code is a length-8, type-II shaping code and the concatenation of the compression and the Varn code is a rate-1, type-I shaping code. The expansion factor of the Varn code is 2.768, which is close to the expansion factor of the optimal type-II shaping code for cost vector $\mathcal{C}'$, where $f_{opt} = 2.737$. Its codeword length distribution is shown in Fig. 3.5.

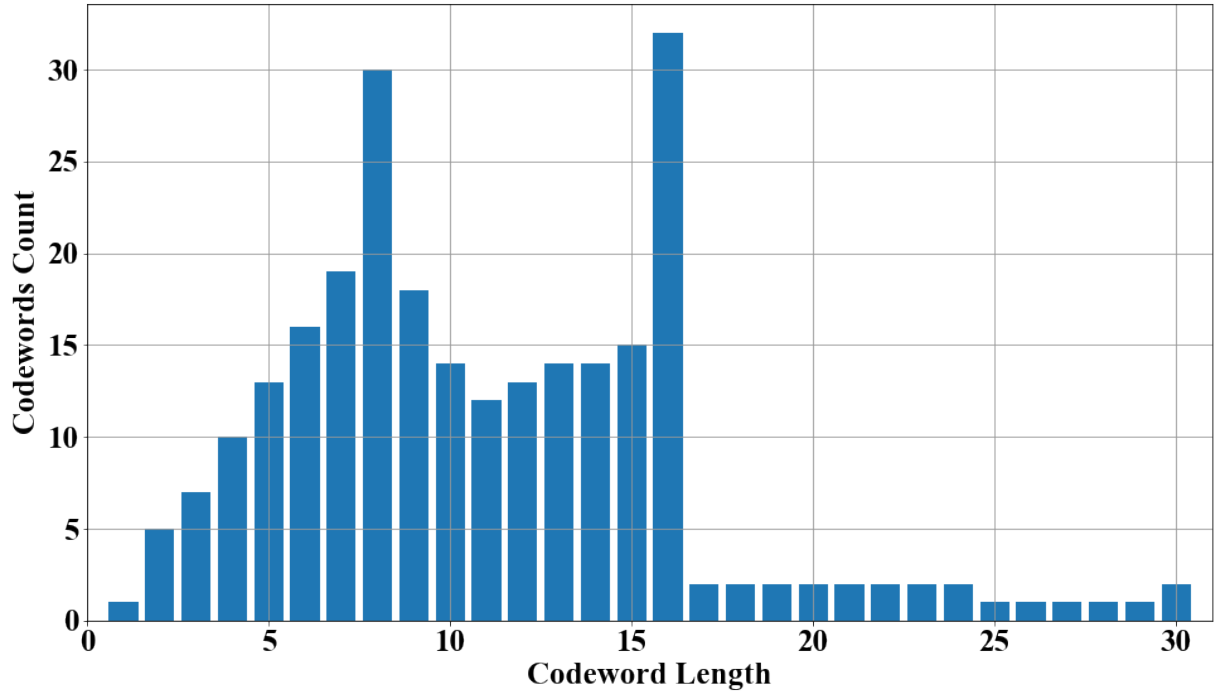


Figure 3.5: Codeword length distribution of Varn code with the codebook size $K = 256$ for English-language text.

To characterize the performance of the designed shaping code, we performed a program/erase (P/E) cycling experiment on the MLC flash memory by repeating the following steps, which collectively represent one P/E cycle. The experiment was conducted with the uncoded source data, and then with the output data from the shaping code.

- Erase the MLC flash memory block.
- Program the MLC flash memory.
- For each successive programming cycle, “rotate” the data, so the data that was written on the i th wordline is written on the $(i + 1)$ st wordline, wrapping around the last wordline to the first wordline.
- After every 100 P/E cycles, erase the block and program pseudo-random data. Then perform a read operation, record bit errors, and calculate the bit error rate.

Fig. 3.6a shows the average bit error rates (BERs) for the uncoded source data, the direct shaping code [44], and the optimal shaping code. The results indicate that the optimal shaping code provides a significant increase in the memory lifetime compared to no shaping and direct shaping.

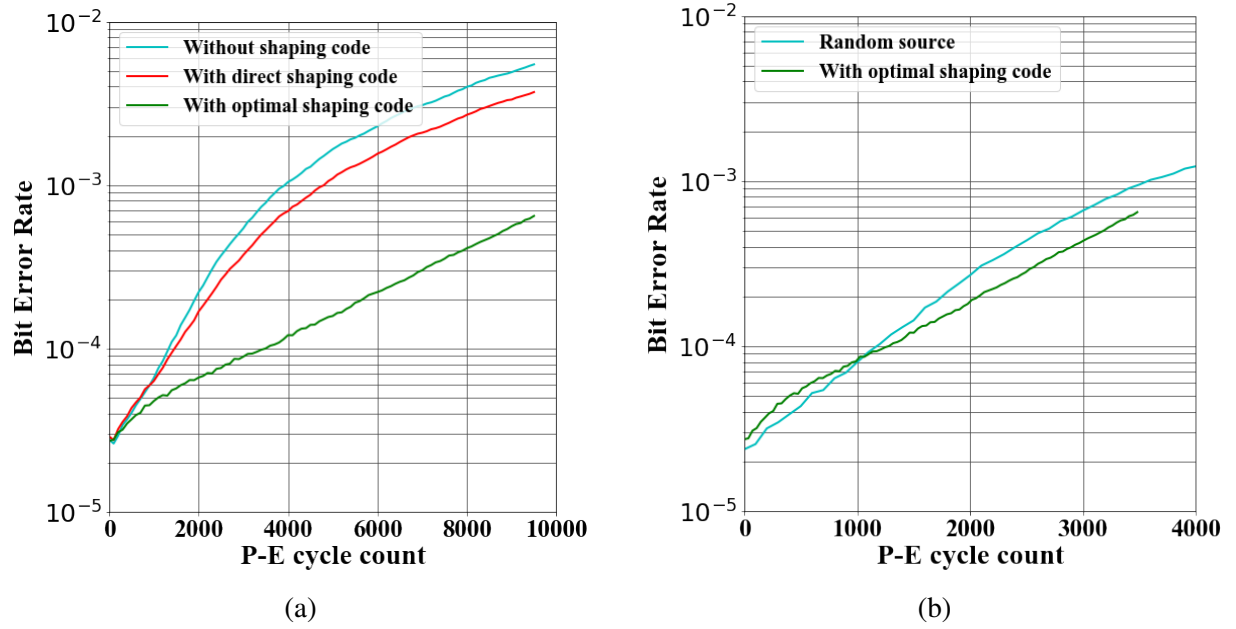


Figure 3.6: BER performance for English-language text.

As a way of comparing the performance of optimal shaping to that of data compression alone, we rescaled the P/E cycle count of the shaping code by the compression ratio 2.740

and compared the result to P/E cycling of pseudo-random data. This corresponds to a BER comparison based upon the total amount of source data stored in the memory. The results, shown in Fig. 3.6b, indicate that the performance of optimal shaping is superior to data compression alone as a function of total source data written.

A similar experiment was conducted for a Chinese-language text, *Collected Works of Lu Xun, Volumes 1–4*, represented using UTF-16LE encoding. We constructed a Varn code with codebook size $K = 256$ based on the cost vector

$$\mathcal{C}' = [0.4222, 2.6647, 3.7860, 5.4099]. \quad (3.163)$$

The expansion factor of the the Varn code was 1.751, which is close to the expansion factor of the optimal type-II shaping code, $f_{opt} = 1.759$. Its codeword length distribution is shown in Fig. 3.7. The BER results are shown in Fig. 3.8a and Fig. 3.8b.

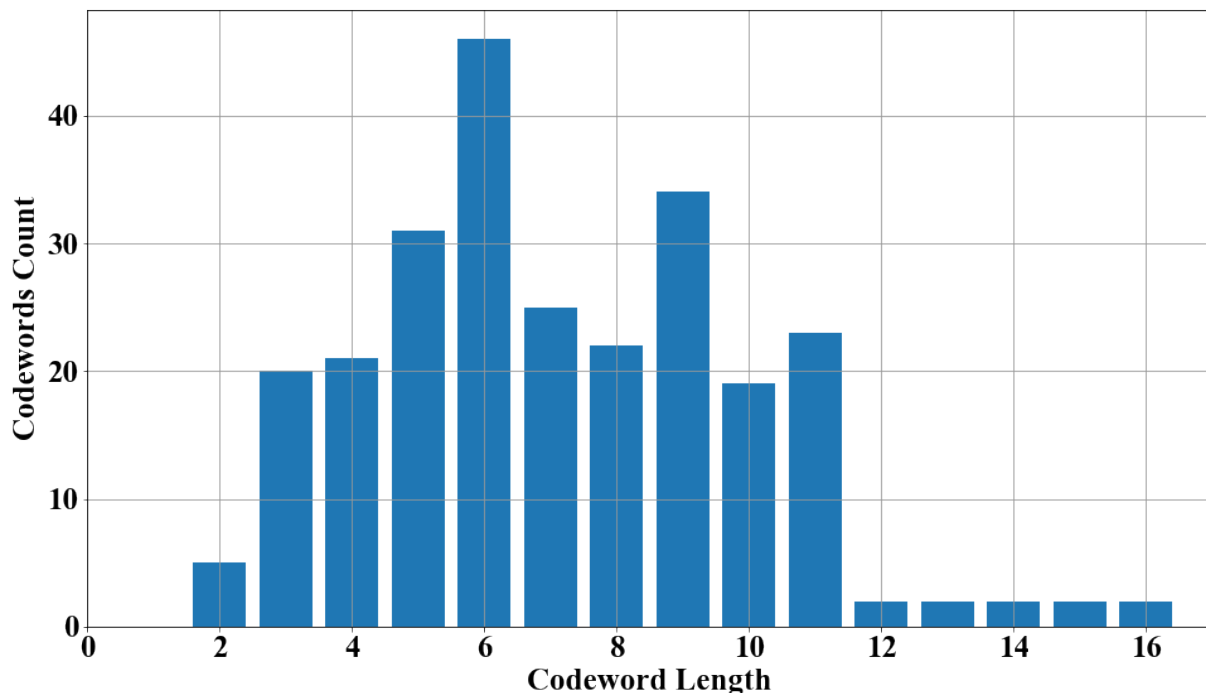


Figure 3.7: Codeword length distribution of Varn code with codebook size $K = 256$ for Chinese-language text.

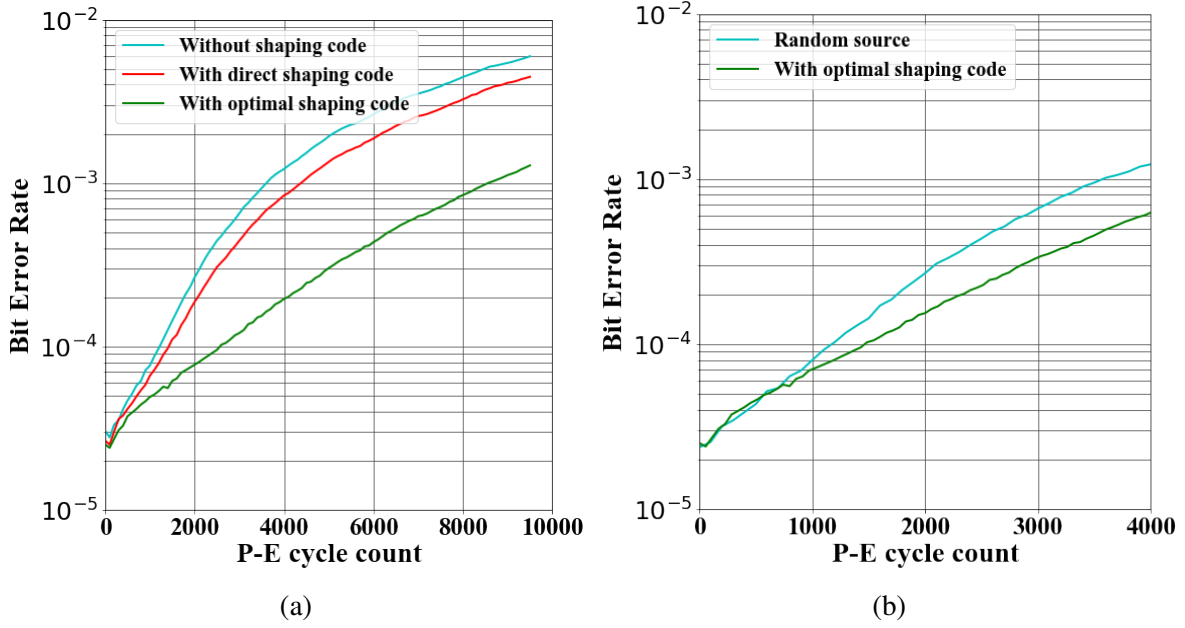


Figure 3.8: BER performance for Chinese-language text.

3.7.2 Varn Codes for Distribution Matching

Remark 3.5.1 shows that the upper bound on the generalized expansion factor of Varn codes decreases as the codebook size increases. This suggests that as the codebook size of a Varn code increases, the approximation to an optimal DM code should improve. In this subsection, we empirically tested this premise by constructing Varn codes with codebook size $K = 100, 1000$ and 10000 , respectively, for a target distribution $\{P_0, P_1\} = \{\frac{2}{3}, \frac{1}{3}\}$. The measure of goodness we used here was similar to the serial test in [60, Section 2.11], namely KL-divergences for patterns of increasing length. Codeword sequences with 10000 codewords were generated using the random number sequence collected from [59]. The first 71514 bits in codeword sequences were used for comparison (71514 was the length of the codeword sequence generated by the Varn code with codebook size $K = 100$). The probability of occurrence of length 1, 2 and 3 patterns was calculated. For example, we define the probability of occurrence of ‘10’ (P'_{10}) and ‘101’ (P'_{101}) in codeword sequence y_1^l as

$$P'_{10} = \frac{\{\text{the number of subsequences } y_i^{i+1} = '10'\}}{l-1}, \quad (3.164)$$

$$P'_{101} = \frac{\{\text{the number of subsequences } y_i^{i+2} = '101'\}}{l-2}. \quad (3.165)$$

The first-, second-, and third-order KL-divergences between P' and distribution $\{P_0, P_1\} = \{\frac{2}{3}, \frac{1}{3}\}$ were calculated, using the following definitions:

$$I_1 = \sum_{i \in \{0,1\}} P'_i \log_2 \frac{P'_i}{P_i} \quad (3.166)$$

$$I_2 = \sum_{i \in \{0,1\}} \sum_{j \in \{0,1\}} P'_{ij} \log_2 \frac{P'_{ij}}{P_i P_j} \quad (3.167)$$

$$I_3 = \sum_{i \in \{0,1\}} \sum_{j \in \{0,1\}} \sum_{k \in \{0,1\}} P'_{ijk} \log_2 \frac{P'_{ijk}}{P_i P_j P_k}. \quad (3.168)$$

The results are shown in Table I. The divergences decrease as K increases, indicating that the approximation to an i.i.d. sequence with target distribution $\{P_0, P_1\} = \{\frac{2}{3}, \frac{1}{3}\}$ is improving.

Table 3.1: First-, second-, and third-order KL-divergence

	P'_0	I_1	I_2	I_3
$K = 100$	0.6447	0.0015	0.0032	0.0055
$K = 1000$	0.6498	0.00091	0.0018	0.0027
$K = 10000$	0.6602	0.00014	0.00027	0.00028

3.8 Conclusion

In this chapter, we studied information-theoretic properties and performance limits of a general class of shaping codes. We determined the asymptotic symbol occurrence probability distribution, and used it to determine the minimum achievable average cost for a type-I shaping code. Using these results, we determined the minimum total cost and optimal expansion factor

for a type-II shaping code. A consequence of this analysis is an equivalence theorem, stating that a type-I shaping code with a given expansion factor and a cost vector can be realized by a type-II shaping code. We then proved a separation theorem stating that optimal shaping can be achieved by a concatenation of optimal lossless compression and optimal shaping for a uniform i.i.d. source. Experimental results showed that optimal shaping can provide a significant increase in flash memory lifetime when applied to English-language and Chinese-language texts, providing total data capacity greater than that achieved by data compression alone.

We also studied properties of prefix-free variable-length distribution matching (DM) codes from the perspective of shaping. We characterized optimal DM codes in terms of the asymptotic normalized divergence and showed that when the divergence equals zero, a DM code encoder generates a codeword sequence that looks i.i.d., with symbol occurrence probability equal to the target distribution. We showed that optimal type-II shaping codes can be used to construct optimal DM codes. This suggested the definition of the generalized expansion factor as a performance measure for DM codes and implied a separation theorem for DM codes. We also established the relationship between the generalized expansion factor and the informational divergence of a DM code. The relationship between the type-I shaping problem and the minimization of normalized informational divergence was also studied. Simulation results showed an increase in distribution matching performance of Varn codes designed for a Bernoulli distribution as the codebook size increases.

3.9 Acknowledgment

Chapter 3 is in part a reprint of the material in the paper: Yi Liu, Pengfei Huang, Alexander W. Bergman, and Paul H. Siegel, Rate-Constrained Shaping Codes for Structured Sources, *IEEE Transactions on Information Theory*, pp. 1-23, Apr. 2020 (Early Access). It is also in part a reprint of materials in the papers: Yi Liu, Pengfei Huang, and Paul H. Siegel, “Performance of

Optimal Data Shaping Codes,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Aachen, Germany, Jun. 25-30, 2017, pp. 1003–1007, Yi Liu, Pengfei Huang, Alexander W. Bergman, and Paul H. Siegel, “Optimal Data Shaping Code Design,” in *Proc. 9th Annu. Non-Volatile Memories Workshop (NVMW)*, La Jolla, CA, Mar. 11-13, 2018, and Yi Liu, Pengfei Huang, and Paul H. Siegel, “Performance of Shaping Codes for Flash Memory,” in *Proc. 8th Annu. Non-Volatile Memories Workshop (NVMW)*, La Jolla, CA, Mar. 12–14, 2017. The dissertation author was the primary investigator and author of these papers.

Chapter 4

Direct Shaping Codes for Multi-Level Flash Memories

4.1 Shaping Codes for SLC Flash

4.1.1 Encoder and Decoder

In the context of SLC flash memory, *direct* shaping codes were first introduced in [67]. Their construction makes use of a rate-1, adaptive encoding dictionary $\mathbf{D}$ that is used to map successive words of length m in the input sequence $\mathbf{W}$ into codewords of length m . Denote by $\mathbf{w}_i \in \{0, 1\}^m$, $i = 1, \dots, 2^m$, the input word and by P_i its probability. Without loss of generality, we assume that $P_1 \geq P_2 \geq \dots \geq P_{2^m}$. The dictionary $\mathbf{D}$ comprises two lists, a dynamically ordered list $\mathcal{X}$ and a fixed output codeword list $\mathcal{Y}$. The output list $\mathcal{Y}$ consists of codewords $\mathbf{y}_k \in \{0, 1\}^m$, $k = 1, 2, \dots, 2^m$, ordered by non-decreasing *cost*, where the cost of a codeword $\mathbf{y}_k$ in this context means the number of 0 symbols in $\mathbf{y}_k$.

At any given time during the encoding process, the list $\mathcal{X}$ consists of pairs $\{(\mathbf{x}_k, n_k)\}$, $k = 1, 2, \dots, 2^m$, where each pair represents a distinct length- m input word $\mathbf{x}_k \in \{0, 1\}^m$ and the *frequency count*, or number of times, n_k , that it has appeared up to that point in the input data

sequence. The list of pairs is dynamically ordered such that the n_k values are in non-increasing order, i.e, $n_1 \geq n_2 \geq \dots \geq n_{2^m}$. At the start of the encoding process, the n_k are all initialized to the value 0, and the words $\mathbf{x}_k$ are ordered lexicographically.

Encoding proceeds as follows. When a data word $\mathbf{x}$ of length m is encountered in the input data sequence, its corresponding pair $(\mathbf{x}_k, n_k) = (\mathbf{x}, n)$ is found in the ordered list $\mathcal{X}$. The encoder then maps $\mathbf{x}$ to the length- m output word $\mathbf{y}$ that occupies the same position in output list $\mathcal{Y}$. The frequency count n of the word $\mathbf{x}$ is increased by 1 and the list $\mathcal{X}$ is reordered accordingly, with the pair $(\mathbf{x}, n+1)$ moved above all pairs with counts less than or equal to $n+1$.

Example 4.1.1. Consider a direct shaping code with parsing length $m = 2$. In the encoding dictionary $\mathbf{D}$, the ordered output list $\mathcal{Y}$ is $\{11, 10, 01, 00\}$. Consider the length-14 data sequence $\mathbf{w} = 10.11.00.10.11.10.00$. The first six length-2 input words contain three words $\mathbf{x}_1 = 10$, two words $\mathbf{x}_2 = 11$ and one word $\mathbf{x}_3 = 00$. The state of the dictionary $\mathbf{D}$ after encoding these 6 words is shown in Table 4.1a.

The final input word is $\mathbf{x} = 00$. According to Table 4.1a, it is mapped to output codeword $\mathbf{y}_3 = 01$. We then add 1 to the count n_3 in the table and update the ordering of the entries in the input list $\mathcal{X}$. The updated dictionary $\mathbf{D}$ is shown in Table 4.1b. □

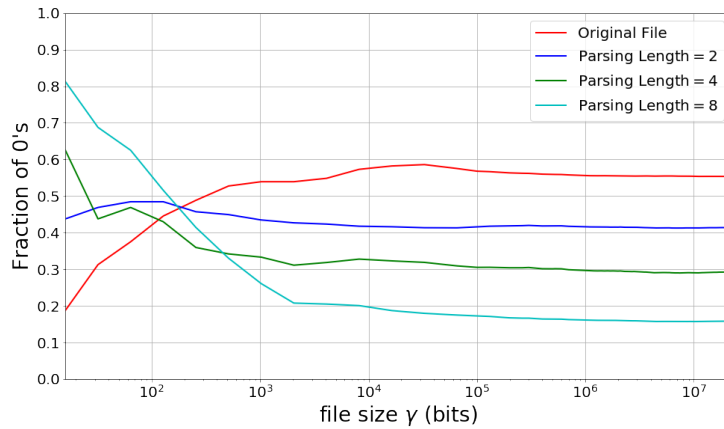


Figure 4.1: Direct shaping codes applied to *The Count of Monte Cristo*.

Table 4.1: Length-2 dictionary when encoding $\mathbf{x} = 00$.

(a)			(b)		
$\mathbf{x}$	n	$\mathbf{y}$	$\mathbf{x}$	n	$\mathbf{y}$
10	3	11	10	3	11
11	2	10	00	2	10
00	1	01	11	2	01
01	0	00	01	0	00

The decoder dynamically reconstructs the table and inverts the encoder mapping. When a codeword $\mathbf{y}$ is encountered in the encoded sequence, it is mapped to the binary length- m data word $\mathbf{x}$ that occupies the same position in the input list $\mathcal{X}$. Then the frequency count n of the word $\mathbf{x}$ is increased by 1 and the ordering of the input list $\mathcal{X}$ is updated accordingly. This rate-1 shaping code incurs no rate penalty and both encoding and decoding can be implemented with low complexity.

4.1.2 Simulation Results

We now present simulation results quantifying the endurance gain which can be achieved by the use of SLC direct shaping codes. The structured data we used was the English-language novel *The Count of Monte Cristo* (TCMC), represented in ASCII Code. We evaluated the shaping code using encoder parsing length m equal to 2, 4 and 8. Fig. 4.1 shows the fraction of 0 symbols in the first γ bits in the original data file and in the corresponding encoded files. The fraction of 0's in the entire original data file was approximately 0.55. With parsing length equals to 2 bits, the fraction of 0's dropped to about 0.41. With parsing length equal to 4 bits, the fraction of 0's dropped to about 0.29, and with parsing length of 8 bits, the fraction was further reduced to about 0.16.

4.2 Data Shaping Codes for MLC Flash

4.2.1 Cost Model for MLC Flash

Every cell in an MLC flash memory can be charged to 4 different values of the threshold voltage, V_{th} . Thus, each cell can represent 2 bits of information. The levels are denoted by 0, 1, 2, 3, respectively, from lowest to highest, and the corresponding binary representations are given by the Gray code {11, 10, 00, 01}. In the binary representation of a level, the left-most bit is called the *lower bit* and the right-most bit is called the *upper bit*. To program the MLC flash cell, we assume the controller uses a two-step process. It first charges the cell to an intermediate voltage level that reflects the value of the lower bit. Then, taking into account the value of the upper bit, it completes charging of the cell to reach the appropriate final threshold voltage level. This process is shown schematically in Fig. 4.2.

The rows of cells in a flash memory block are called *wordlines*, and the wordlines are programmed sequentially, in a row-by-row manner. The lower bits of cells in a wordline constitute the *lower page*, while the upper bits form the *upper page*. When programming a block, information is programmed separately to lower pages and upper pages. During data retrieval, pages are also retrieved independently, with lower bit values recovered by reference to read threshold V_B , and upper bit values by reference to read thresholds V_A and V_C , also shown in Fig. 4.2.

To characterize and quantify the damage caused by different programmed levels in an MLC flash memory, we performed an experiment on several blocks in which we repeatedly programmed all the cells in the block to a specified, constant level, erasing the block (i.e., reducing voltage to level 0) after each programming cycle. The device we used was a 1x nm MLC flash chip with the default read threshold positions. After every 100 of these P/E cycles, we programmed the block with pseudo-random data, read it back, and recorded the cell error rate. The error rates, averaged over the blocks used in the experiment, are shown in Fig. 4.3. We see

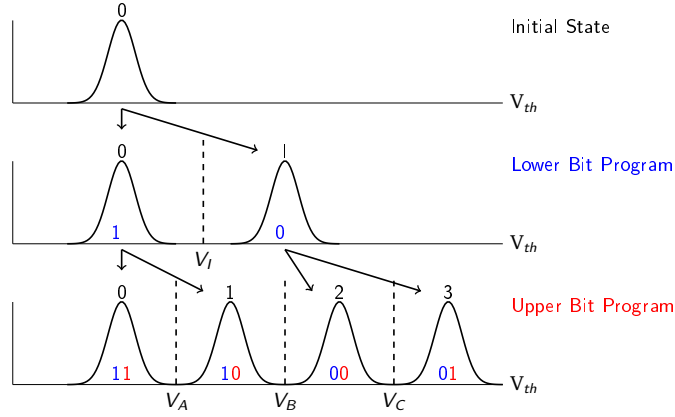


Figure 4.2: Schematic of MLC flash cell programming.

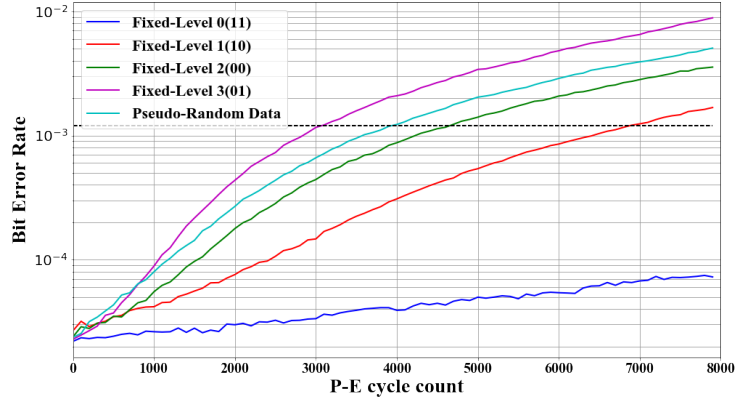


Figure 4.3: Cell error rate of MLC Flash for different programming levels, while repeatedly programming a constant level.

that cell damage caused by the levels 0, 1, 2, 3 increases monotonically. A *cost model*, in the form of a vector of cell-level costs $\mathcal{C} = [c_0, c_1, c_2, c_3]$, is used to quantify the relative amount of device wear associated with each of the cell levels. In practice, these costs will satisfy $c_0 \leq c_1 \leq c_2 \leq c_3$, reflecting the increased damage induced by higher programmed levels.

Given a length- m cell-level codeword $\mathbf{z} = [z_1, \dots, z_m]$, we denote the cost associated with the symbol z_i by $c(z_i)$, and the total cost $c(\mathbf{z})$ associated with programming $\mathbf{z}$ is assumed to be the sum of the individual symbol costs; i.e., $C = c(\mathbf{z}) = \sum_{i=1}^m c(z_i)$.

Remark 4.2.1. A method for determining physically meaningful values of c_i was proposed by Li

et al. in [42]. It is based upon measuring the number of cycles it takes for the error rate associated with a certain programmed level to reach a prespecified *maximum tolerable cell error rate*. If the design lifetime of the flash memory is T_0 P/E cycles, the maximum tolerable cell error rate CER_{max} is defined as the cell error rate after the memory is programmed with random data for T_0 P/E cycles. We view the damage caused by programming random data to the memory as being proportional to $1/T_0$.

We define $\Phi^i(T)$ to be the cell error rate observed after T cycles of programming the blocks to the level i , for $i = 0, 1, 2, 3$. Let T_{max}^i denote the P/E cycle number at which $\Phi^i(T)$ equals CER_{max} . We view the damage caused by programming to level i as proportional to $1/T_{max}^i$, so we define the cost associated with each level i , $i = 0, 1, 2, 3$, to be

$$c_i = \frac{T_0}{T_{max}^i}. \quad (4.1)$$

We calculate the cost model corresponding to the error rate results in Fig. 4.3, assuming a design lifetime of $T_0 = 4000$ cycles. The error rate associated with level 0 remains essentially constant, so we set $c_0 = 0$. For the other levels, we find that $T_{max}^1 = 6900$, $T_{max}^2 = 4600$ and $T_{max}^3 = 3100$, so the complete cost model is computed to be $[0, 0.58, 0.87, 1.29]$. $\square$

4.2.2 Shaping Codes for MLC Flash

Application of the SLC shaping code independently to lower and upper pages will not be effective in reducing the average cost. This can be seen by referring to the schematic of the MLC flash cell programming process in Fig. 4.2. As shown in the schematic, the cost associated with an upper bit 1 or 0 depends on the value of the corresponding lower bit in the cell. The proposed MLC shaping encoder achieves improved wear reduction by using lower-page dependent dictionaries when encoding the upper pages, as we now describe.

First, fix a parsing length m . Encoding and programming of wordlines is done in a

row-by-row, sequential manner. Suppose that we want to encode L data words to both lower and upper pages. For the lower pages, the encoder simply uses the direct shaping code with parsing length n . When programming an upper page, however, the encoder first reads the corresponding, previously programmed lower page.

Now, suppose the lower page that has been programmed and presumably correctly recovered consists of the sequence of length- m codewords $\mathbf{v}^{(k)}$, $k = 0, \dots, L$. Consider the sequence of length- m data words $\mathbf{w}^{(k)}$, $k = 0, \dots, L$ that need to be encoded for the upper page. To encode the k th data word $\mathbf{w}^{(k)}$, we encode using a direct shaping code that is based upon an adaptively-built dictionary that depends on the corresponding lower page codeword $\mathbf{v}^{(k)}$. The only difference in the operation of each lower-page dependent shaping encoder is that the ordering of the length- m encoder output words in terms of increasing cost depends specifically on the lower page codeword $\mathbf{v}$ and the cost model $[c_0, c_1, c_2, c_3]$.

To illustrate the design of the encoder, we will use the cost model $[c_0, c_1, c_2, c_3] = [0, 1, 1, 2]$. This simple cost model is motivated as follows. Consider the standard lower-upper page binary representation of the cell levels: 0=11, 1=10, 2=00, 3=01. We assign a cost of 0 to lower bit value 1, and a cost of 1 to lower bit value 0, in accordance with the MLC cell programming schematic. When the lower bit value is 1, we assign a cost of 0 to upper bit value 1, and a cost of 1 to upper bit value 0. On the other hand, when the lower bit value is 0, we assign a cost of 0 to upper bit value 0, and a cost of 1 to upper bit value 1. Again, these cost assignments are consistent with the MLC cell programming schematic. The cost associated with a cell level is then defined as the sum of the corresponding lower bit and upper bit costs.

Choose parsing length $n = 4$ and assume the lower page codeword is $\mathbf{v} = \mathbf{1110}$. For the first three bits, where the corresponding lower bit is a 1, programming the upper bit to 1 is better than to 0. For the last bit, where the corresponding lower bit is a 0, programming the upper bit to 0 is better than to 1. This implies that the lowest cost output word in the dictionary is 1110, corresponding to cell level word 0002, which has total cost 1. Similar reasoning leads to the

Table 4.2: Ordered list of upper page words for lower page codeword 1110.

index	Output List	index	Output List
0	1110	8	1000
1	1111	9	0100
2	1100	10	0010
3	1010	11	1001
4	0110	12	0101
5	1101	13	0011
6	1011	14	0000
7	0111	15	0001

Table 4.3: Corresponding list of cell level words for lower page codeword 1110.

index	Output List	index	Output List
0	0002	8	0112
1	0003	9	1012
2	0012	10	1102
3	0102	11	0113
4	1002	12	1013
5	0013	13	1103
6	0103	14	1112
7	1003	15	1113

ordered list of output words shown in Table 4.2. The corresponding list of cell level words is given in Table 4.3.

It is easy to verify that the costs of the corresponding cell level words are non-decreasing: cost 2 for words 1 to 4, cost 3 for words 5 to 10, cost 4 for words 11 to 14, and cost 5 for word 15. In general, the encoder uses 2^m direct shaping encoders operating in a sequence determined by the sequence of lower page codewords $\mathbf{v}^{(k)}$, $k = 0, \dots, L$.

4.2.3 Encoding Algorithm for MLC Shaping Codes

The upper page encoder uses an encoding table in which the order of the output codewords depends on the lower page codeword. The encoding table provides a mapping from an index $I \in$

$\{1, \dots, 2^m\}$ to the corresponding upper page output word $\mathbf{y}$ in the ordered list. This mapping can be implemented using enumerative coding, introduced by Cover [12]. To explain the algorithm, we first introduce some notation. We denote the length- m lower page codeword by $\mathbf{v}$ and the upper page word by $\mathbf{y}$. They determine the cell level output word $\mathbf{z} = [z_1, \dots, z_m]$, where z_i is the cell level associated with the lower and upper bit pair $v_i y_i$.

Let $n(\mathbf{v}, C, y_1, y_2, \dots, y_k)$ be the number of upper page words $\mathbf{y}$ that, together with lower page codeword $\mathbf{v}$, yield total cost C , and whose first k coordinates are equal to $[y_1, y_2, \dots, y_k]$. To determine $n(\mathbf{v}, C, y_1, y_2, \dots, y_k)$, we first calculate the polynomial $g_{\mathbf{v},k}(x) = (x^{c_0} + x^{c_1})^{n_1} (x^{c_2} + x^{c_3})^{n_0}$, where $n_b, b \in \{0, 1\}$ is the number of bits equal to b in the last $m - k$ bits of the lower page codeword, $[v_{k+1}, \dots, v_m]$. The degrees of terms in $g_{\mathbf{v},k}(x)$ represent the possible total costs of cell level vectors $[z'_{k+1}, \dots, z'_m]$ with associated lower bit vector $[v_{k+1}, \dots, v_m]$, and the corresponding coefficients represent the number of such vectors. Then $n(\mathbf{v}, C, y_1, y_2, \dots, y_k)$ is equal to the coefficient of the term in $g_{\mathbf{v},k}(x)$ whose degree is equal to $C - \sum_{i=1}^k u_i$. If there is no such term in the polynomial, we set $n(\mathbf{v}, C, y_1, y_2, \dots, y_k) = 0$. In particular, the polynomial $g_{\mathbf{v},0}(x)$ corresponding to $[v_1, \dots, v_m]$ tells us the costs $C_j, j = 1, \dots, J$ of all possible length- m cell level vectors with associated lower bit vector $\mathbf{v}$, along with the number of upper page words that together with $\mathbf{v}$ produce these costs. From these we calculate $n(\mathbf{v}, C)$, the number of upper page words that produce total cost less than or equal to U , given lower page codeword $\mathbf{v}$. Example 4.2.1 illustrates how to calculate $n(\mathbf{v}, C, y_1, y_2, \dots, y_k)$.

Example 4.2.1. We want to calculate $n(\mathbf{v}, C, 1, 1)$ where the lower page codeword is $\mathbf{v} = \mathbf{1110}$ and the total cost is $C = 2$. The combined cost of the first two cells is 0, so the remaining cost is 2. We calculate the polynomial $(x^{c_0} + x^{c_1})^{n_1} (x^{c_2} + x^{c_3})^{n_0} = (1 + x)(x + x^2) = x + 2x^2 + x^3$. The coefficient of x^2 is 2, meaning there are two possible codewords with total cost $C = 2$ and first two bits $[1, 1]$. $\square$

With enumerative coding, instead of storing the ordered upper page output word list, we only need to store the values $n(\mathbf{v}, C, y_1, y_2, \dots, y_k)$ and $n(\mathbf{v}, C)$. Algorithm 1 describes the

encoding process.

Algorithm 1 Encoding upper page codewords

Require: Codeword length m , lower page codeword $\mathbf{v}$, index I

Ensure: Upper page codeword $\mathbf{y} = (y_1, y_2, \dots, y_m)$

- 1: Find j such that $I > n(\mathbf{v}, C_{j-1})$ and $I \leq n(\mathbf{v}, C_j)$, set $I = I - n(\mathbf{v}, C_{j-1})$
 - 2: If $I > n(\mathbf{v}, C_j, 0)$ set $y_1 = 1$ and set $I = I - n(\mathbf{v}, C_j, 0)$, otherwise set $y_1 = 0$
 - 3: For i th bit, if $I > n(\mathbf{v}, C_j, y_1, \dots, y_{i-1}, 0)$ set $y_i = 1$ and set $I = I - n(\mathbf{v}, C_j, y_1, \dots, y_{i-1}, 0)$, otherwise set $y_i = 0$
-

4.2.4 Simulation Results

We assessed the performance of the MLC shaping encoder using TCMC. The data file was divided into two consecutive subfiles of size 11,134,796 bits; the first was used for lower page encoding and the second for upper page encoding.

Fig. 4.4 shows the respective fractions of the encoded cell levels 0, 1, 2, 3 appearing in the cell level sequences produced when no shaping code was applied to the first γ bits of each subfile. Fig. 4.5 shows the corresponding fractions when the proposed MLC shaping encoder is applied using the empirical cost model derived in Remark. 4.2.1.

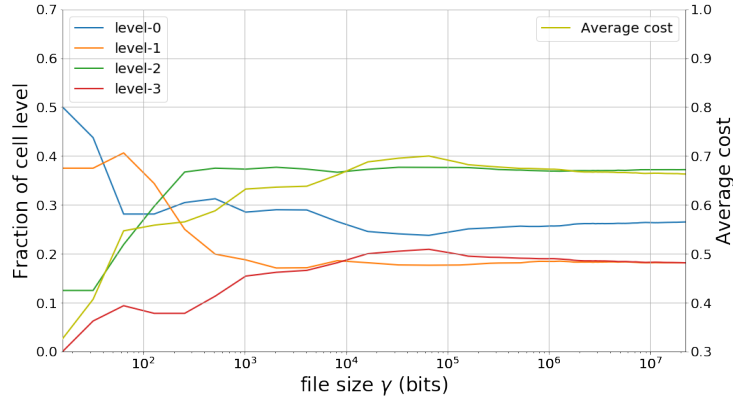


Figure 4.4: Fractions of MLC cell levels for segments of *The Count of Monte Cristo* without a shaping code.

From these figures, we see that for TCMC, the average cost when no coding is applied is

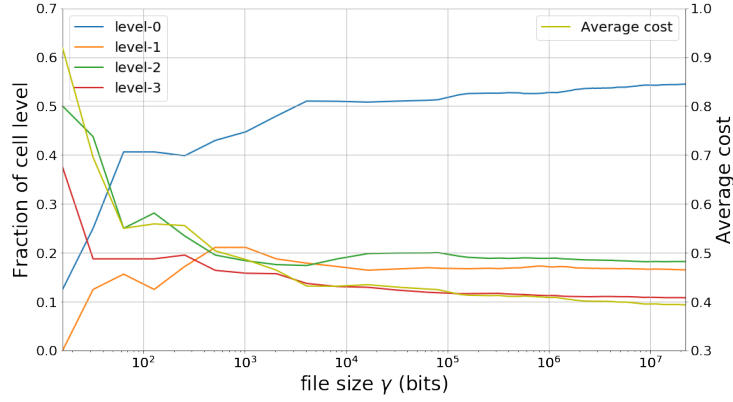


Figure 4.5: Fractions of MLC cell levels for segments of *The Count of Monte Cristo* with MLC shaping code.

0.66. When SLC shaping is applied independently to lower and upper pages, the average cost is 0.48 (not shown). In contrast, after MLC shaping, the average cost is reduced to 0.39.

4.2.5 Experimental Results on MLC Flash Memory

In this subsection, we present the experimental results that evaluate the performance of MLC shaping codes. To characterize the performance of the shaping code, we performed a program/erase (P/E) cycling experiment on the MLC flash memory by repeating the following steps, which collectively represent one P/E cycle.

- Erase the MLC flash memory block.
- Program the MLC flash memory.
- For each successive programming cycle, “rotate” the data, so the data that was written on the i th wordline is written on the $i + 1$ st wordline, wrapping around the last wordline to the first wordline.
- After every 100 P/E cycles, erase the block and program pseudo-random data. Then perform a read operation, record bit errors, and calculate the bit error rate.

The experiment was conducted with the uncoded source data, and then with the output data from the MLC shaping code described in Section 4.2.4. The average bit error rates (BERs) are shown in Fig. 4.6. The black dash line represents the maximum tolerable cell error rate introduced in Remark 4.2.1 and the increased lifetime was measured at this rate. The results indicate that applying MLC shaping codes to English text increases the lifetime of flash memory device by 900 P/E cycles. A similar experiment was done using the Chinese text *Collected Works of Lu Xun, Volumes 1–4*. Fig. 4.7 shows that MLC shaping codes increase the lifetime of flash memory device by 800 P/E cycles.

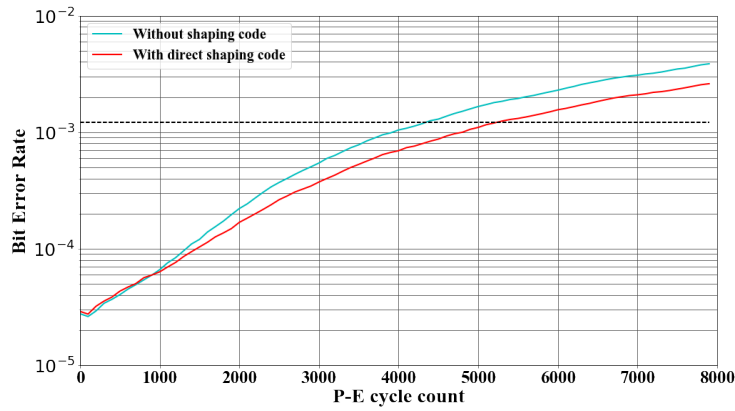


Figure 4.6: BER performance of English-language text *The Count of Monte Cristo*.

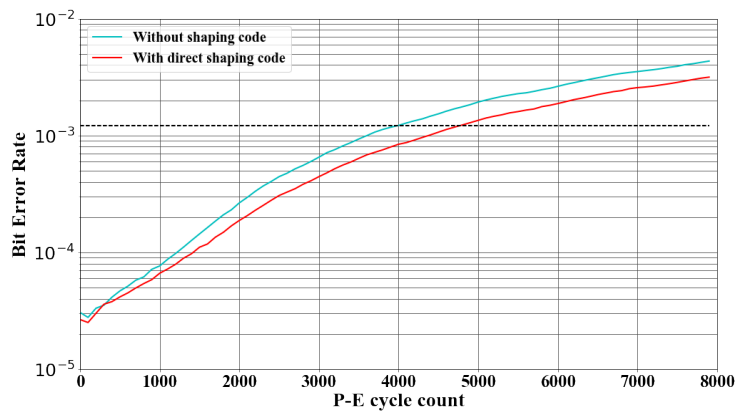


Figure 4.7: BER performance of Chinese-language text *Collected Works of Lu Xun, Volumes 1–4*.

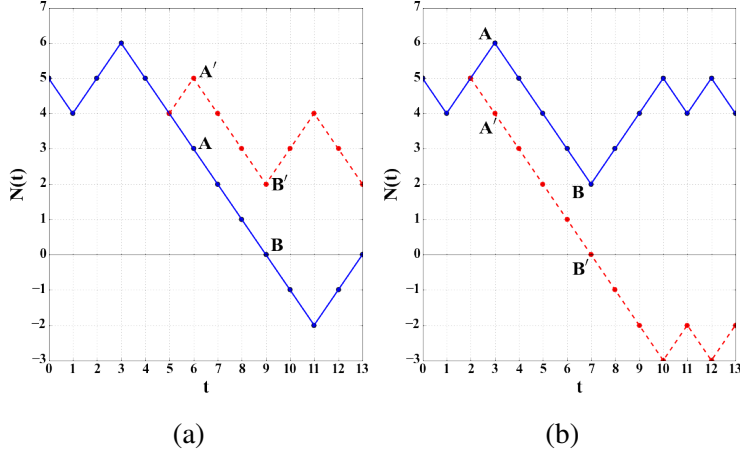


Figure 4.8: Error propagation examples.

4.3 Error Propagation Analysis

The direct shaping decoder reproduces the dynamic construction of the encoding list $\mathcal{X}$. Errors in reading the flash memory can lead to incorrect word frequency counts that, in turn, can cause decoding errors if word counts are not sufficiently separated. In this section we introduce a framework for analyzing potential error propagation properties of direct shaping codes, based upon recurrence properties of random walks.

4.3.1 Recurrence Probability of a Two-Word Dictionary

We first consider the case when $m = 1$, where the input dictionary contains only two words $\mathbf{w}_1$ and $\mathbf{w}_2$. Suppose that, at time t_0 , the input word counts satisfy $n_1(t_0) \neq n_2(t_0)$. Since there are only 2 words, the dictionary after time t can be represented by the word that has the higher count, denoted by $s(t) \in \{\mathbf{w}_1, \mathbf{w}_2\}$, and the distance, denoted simply by $N(t) = n_1(t) - n_2(t)$. Let $s^e(t), N^e(t)$ represent the dictionary evolution during the encoding process, and let $s^d(t), N^d(t)$ represent the dictionary evolution during decoding process. The distance behaves like a one-dimensional random walk, as depicted in Fig. 4.8a. In the figure, the blue line represents the encoding process, starting at time denoted as $t = 0$, with $s^e(0) = \mathbf{w}_1$ and $N^e(0) = 5$.

Table 4.4: Encoding and decoding processes in Fig. 4.8a.

t	0	1	2	3	4	5	6	7	8	9	10	11	12	13
(s^e, N^e)	$(\mathbf{w}_1, 5)$	$(\mathbf{w}_1, 4)$	$(\mathbf{w}_1, 5)$	$(\mathbf{w}_1, 6)$	$(\mathbf{w}_1, 5)$	$(\mathbf{w}_1, 4)$	$(\mathbf{w}_1, 3)$	$(\mathbf{w}_1, 2)$	$(\mathbf{w}_1, 1)$	$(\mathbf{w}_2, 0)$	$(\mathbf{w}_2, -1)$	$(\mathbf{w}_2, -2)$	$(\mathbf{w}_2, -1)$	$(\mathbf{w}_1, 0)$
input	—	$\mathbf{w}_2$	$\mathbf{w}_1$	$\mathbf{w}_1$	$\mathbf{w}_2$	$\mathbf{w}_2$	$\mathbf{w}_2$	$\mathbf{w}_2$	$\mathbf{w}_2$	$\mathbf{w}_2$	$\mathbf{w}_2$	$\mathbf{w}_2$	$\mathbf{w}_1$	$\mathbf{w}_1$
output	—	0	1	1	0	0	0	0	0	0	1	1	0	0
(s^d, N^d)	$(\mathbf{w}_1, 5)$	$(\mathbf{w}_1, 4)$	$(\mathbf{w}_1, 5)$	$(\mathbf{w}_1, 6)$	$(\mathbf{w}_1, 5)$	$(\mathbf{w}_1, 4)$	$(\mathbf{w}_1, 5)$	$(\mathbf{w}_1, 4)$	$(\mathbf{w}_1, 3)$	$(\mathbf{w}_1, 2)$	$(\mathbf{w}_1, 3)$	$(\mathbf{w}_1, 4)$	$(\mathbf{w}_1, 3)$	$(\mathbf{w}_1, 2)$
input	—	0	1	1	0	0	1	0	0	0	1	1	0	0
output	—	$\mathbf{w}_2$	$\mathbf{w}_1$	$\mathbf{w}_1$	$\mathbf{w}_2$	$\mathbf{w}_2$	$\mathbf{w}_1$	$\mathbf{w}_2$	$\mathbf{w}_2$	$\mathbf{w}_2$	$\mathbf{w}_1$	$\mathbf{w}_1$	$\mathbf{w}_2$	$\mathbf{w}_2$

Assuming no read errors through time $t = 5$, the decoder correctly reconstructs the encoding dictionary, retracing the blue line. At time $t = 5$, we have $s^d(5) = s^e(5) = \mathbf{w}_1$ and $N^e(5) = N^d(5) = 4$. At $t = 6$, the input was $\mathbf{w}_2$, implying $s^e(6) = \mathbf{w}_1$ and $N^e(6) = 3$, producing output codeword 0. This point is marked by A on the blue line. Now, suppose a read error occurs at decoding time $t = 6$. The codeword is incorrectly read as 1, which is decoded as input $\mathbf{w}_1$, resulting in $s^d(6) = \mathbf{w}_1$ and $N^d(6) = 5$. This deviation is marked by A' on the red dashed line, which represents the decoding process. Assume no further read errors occur. The decoder trace remains separated from the encoder trace, but the following three codewords are nevertheless decoded correctly. At time $t = 9$, since $N^e(9) = 0$, the encoder changes to $s^e(9) = \mathbf{w}_2$, while $N^d(9) = 2$ implies that the decoder continues with $s^d(9) = \mathbf{w}_1$. These two points are indicated by B and B' on the encoding and decoding traces, respectively. The next input word was $\mathbf{w}_2$ and the encoder output was 1, but, even though it is read correctly, the decoded word is $\mathbf{w}_1$. In fact, after this point, all codewords will be decoded incorrectly. This example shows that if the encoding dictionary reaches a point where $N^e(t) = 0$, there is the potential for error propagation. Fig. 4.8b also shows the potential of error propagation if the decoding dictionary reaches a point where $N^d(t) = 0$. The encoding and decoding processes in Fig. 4.8 are summarized in Tables 4.4 and 4.5.

In order to analyze the error propagation behavior of direct shaping codes, we make use of a result about one-dimensional random walks. Here we use the encoding process as an example. For a two-word dictionary with input words $\mathbf{w}_1$ and $\mathbf{w}_2$ and respective counts $n_1^e(t)$ and $n_2^e(t)$, suppose $n_1^e(t_0) \neq n_2^e(t_0)$ at time t_0 . We say a *recurrence* occurs if, at some future time $t > t_0$,

Table 4.5: Encoding and decoding processes in Fig. 4.8b.

t	0	1	2	3	4	5	6	7	8	9	10	11	12	13
(s^e, N^e)	$(\mathbf{w}_1, 5)$	$(\mathbf{w}_1, 4)$	$(\mathbf{w}_1, 5)$	$(\mathbf{w}_1, 6)$	$(\mathbf{w}_1, 5)$	$(\mathbf{w}_1, 4)$	$(\mathbf{w}_1, 3)$	$(\mathbf{w}_1, 2)$	$(\mathbf{w}_1, 3)$	$(\mathbf{w}_1, 4)$	$(\mathbf{w}_1, 5)$	$(\mathbf{w}_1, 4)$	$(\mathbf{w}_1, 5)$	$(\mathbf{w}_1, 4)$
input	—	$\mathbf{w}_2$	$\mathbf{w}_1$	$\mathbf{w}_1$	$\mathbf{w}_2$	$\mathbf{w}_2$	$\mathbf{w}_2$	$\mathbf{w}_2$	$\mathbf{w}_1$	$\mathbf{w}_1$	$\mathbf{w}_1$	$\mathbf{w}_2$	$\mathbf{w}_1$	$\mathbf{w}_2$
output	—	0	1	1	0	0	0	0	1	1	1	0	1	0
(s^d, N^d)	$(\mathbf{w}_1, 5)$	$(\mathbf{w}_1, 4)$	$(\mathbf{w}_1, 5)$	$(\mathbf{w}_1, 4)$	$(\mathbf{w}_1, 3)$	$(\mathbf{w}_1, 2)$	$(\mathbf{w}_1, 1)$	$(\mathbf{w}_2, 0)$	$(\mathbf{w}_2, -1)$	$(\mathbf{w}_2, -2)$	$(\mathbf{w}_2, -3)$	$(\mathbf{w}_2, -2)$	$(\mathbf{w}_1, -3)$	$(\mathbf{w}_1, -2)$
input	—	0	1	0	0	0	0	0	1	1	1	0	1	0
output	—	$\mathbf{w}_2$	$\mathbf{w}_1$	$\mathbf{w}_2$	$\mathbf{w}_2$	$\mathbf{w}_2$	$\mathbf{w}_2$	$\mathbf{w}_2$	$\mathbf{w}_2$	$\mathbf{w}_2$	$\mathbf{w}_2$	$\mathbf{w}_1$	$\mathbf{w}_2$	$\mathbf{w}_1$

$n_1^e(t) = n_2^e(t)$. The following theorem, stated in [15, Theorem 4.8.9], determines the recurrence probability.

Theorem 4.3.1. *Consider two input words $\mathbf{w}_1, \mathbf{w}_2$, with probabilities P_1, P_2 respectively ($P_1 + P_2 = 1$, $P_1 \geq P_2$). Let $N^e(t) = n_1^e(t) - n_2^e(t)$. At time t_0 , the probability Q that a recurrence involving $\mathbf{w}_1, \mathbf{w}_2$ will occur in the future is*

$$Q = \begin{cases} \left(\frac{P_2}{P_1}\right)^{N^e(t_0)} & \text{if } N^e(t_0) > 0, \\ 1 & \text{if } N^e(t_0) \leq 0. \end{cases} \quad (4.2)$$

□

Here we assume that the SLC flash memory is a binary symmetric channel with transition probability ρ . The pair $\{N^e(t), N^d(t)\}$ acts like a two-dimensional random walk, as shown in Fig. 4.9. The transition probabilities of this random walk are

$$\begin{aligned} P(\{N^e(t), N^d(t)\} \rightarrow \{N^e(t) + 1, N^d(t) + 1\}) &= (1 - \rho)P_1, \\ P(\{N^e(t), N^d(t)\} \rightarrow \{N^e(t) + 1, N^d(t) - 1\}) &= \rho P_1, \\ P(\{N^e(t), N^d(t)\} \rightarrow \{N^e(t) - 1, N^d(t) - 1\}) &= (1 - \rho)P_2, \\ P(\{N^e(t), N^d(t)\} \rightarrow \{N^e(t) - 1, N^d(t) + 1\}) &= \rho P_2. \end{aligned} \quad (4.3)$$

We say a recurrence occurs if, at future time $t > t_0$, either $N^e(t)$ or $N^d(t)$ equals to zero. We denote by E^e the event that the recurrence occurs during the encoding process (at future time $t > t_0$, $N^e(t)$ equals to zero) and E^d the event that the recurrence occurs during the decoding

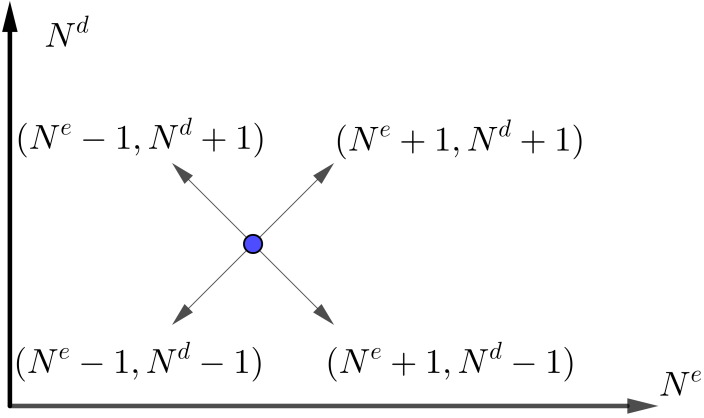


Figure 4.9: Two dimensional random walk on (N_i^e, N_i^d) -plane

process. The recurrence probability is

$$Q = \Pr\{E^e \cup E^d\}. \quad (4.4)$$

Using Theorem 4.3.1, we can calculate $\Pr\{E^e\}$ and $\Pr\{E^d\}$. When $\rho \geq 0.5$, $N^d(t)$ increases by 1 with probability

$$\begin{aligned} P_d &= (1 - \rho)P_1 + \rho P_2 \\ &= (1 - \rho)P_1 + \rho(1 - P_1) \\ &= \frac{1}{2} - \frac{1}{2}(1 - 2\rho)(1 - 2P_1) \leq \frac{1}{2}. \end{aligned} \quad (4.5)$$

This indicates that $\Pr\{E^d\} = 1$ when $N^d(t) > 0$. Similarly, when $N^e(t) < 0$, from Theorem 4.3.1 we have $\Pr\{E^e\} = 1$. To avoid these situations, we assume that $\rho < 0.5$ and $N^e(t_0), N^d(t_0) > 0$ at time t_0 . we say the pair $\{N^e(t_0), N^d(t_0)\}$ is *stable* if $N^e(t_0), N^d(t_0) > 0$. The following theorem provides an upper bound on Q .

Theorem 4.3.2. *For a two-word dictionary with a stable pair $\{N^e(t_0), N^d(t_0)\}$ at time t_0 , the probability Q that a recurrence will occur in the future is bounded by*

$$Q \leq \left(\frac{P_2}{P_1}\right)^{N^e(t_0)} + \left[\frac{\rho P_1 + (1 - \rho)P_2}{(1 - \rho)P_1 + \rho P_2}\right]^{N^d(t_0)}. \quad (4.6)$$

Proof. A simple union bound yields

$$Q = \Pr\{E^e \cup E^d\} \leq \Pr\{E^e\} + \Pr\{E^d\}. \quad (4.7)$$

From Theorem 4.3.1, we have

$$\Pr\{E^e\} = \left(\frac{P_2}{P_1}\right)^{N^e(t_0)} \quad (4.8)$$

and

$$\Pr\{E^d\} = \left[\frac{\rho P_1 + (1 - \rho)P_2}{(1 - \rho)P_1 + \rho P_2}\right]^{N^d(t_0)}. \quad (4.9)$$

Combining equations (4.7), (4.8) and (4.9) completes the proof. $\square$

4.3.2 Recurrence Probability of a Pair of Words in a General Dictionary

In this subsection, we study the recurrence probability of a pair of words in the general dictionary with 2^m entries. We first establish some notation. Assume t data words have been encoded. Let $\mathbf{n}^e(t) = \{n_1^e(t), n_2^e(t), n_3^e(t), \dots, n_{2^m}^e(t)\}$ be the word counts and let $N_i^e(t) = n_i^e(t) - n_{i+1}^e(t)$ denote the *distance* between the i th and $(i + 1)$ st words. Similarly, for the decoding process, let $\mathbf{n}^d(t) = \{n_1^d(t), n_2^d(t), \dots, n_{2^m}^d(t)\}$ be the word counts and let $N_i^d(t) = n_i^d(t) - n_{i+1}^d(t)$ denote the distance. We say the pair $\{\mathbf{n}^e(t), \mathbf{n}^d(t)\}$ is *stable* if $n_1^e(t) > n_2^e(t) > \dots > n_{2^m}^e(t)$ and $n_1^d(t) > n_2^d(t) > \dots > n_{2^m}^d(t)$. We sometimes use the term *stable* to describe a dictionary that has a stable $\{\mathbf{n}^e(t), \mathbf{n}^d(t)\}$. When the dictionary is stable, the probability of reading word $\mathbf{w}_i$ from flash memory is

$$P_i^d = \sum_j \rho^{d_{\{i,j\}}} (1 - \rho)^{m - d_{\{i,j\}}} P_j, \quad (4.10)$$

where $d_{\{i,j\}}$ is the Hamming distance between $\mathbf{w}_i$ and $\mathbf{w}_j$. Here we always assume that ρ is small enough so that $P_1^d \geq P_2^d \geq \dots \geq P_{2^m}^d$.

We assume that the dictionary is stable at some t_0 . We denote by $\{i, i + 1\}^e$ the event that a recurrence occurs between the i th and $(i + 1)$ st words during the encoding process and

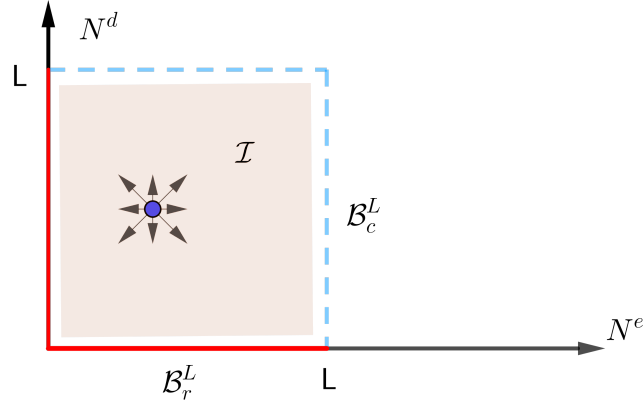


Figure 4.10: Two dimensional random walk in a general dictionary

by $\{\overline{i, i+1}\}^e$ its complement. Similarly, we denote by $\{i, i+1\}^d$ the event that a recurrence occurs between the i th and $(i+1)$ st words during the decoding process and by $\{\overline{i, i+1}\}^d$ its complement. In this subsection, we study the recurrence probability $Q_i(N_i^e(t), N_i^d(t)) = \Pr\{\{i, i+1\}^e \cup \{i, i+1\}^d\}$. We first note the following lemma which is a direct corollary of Theorem 4.3.2.

Lemma 4.3.3. *For a general dictionary with a stable pair $\{N_i^e(t_0), N_i^d(t_0)\}$ at time t_0 , the probability $Q_i(N_i^e(t_0), N_i^d(t_0))$ that a recurrence will occur in the future between $\mathbf{w}_i$ and $\mathbf{w}_{i+1}$ is bounded by*

$$Q_i(N_i^e(t_0), N_i^d(t_0)) \leq \left(\frac{P_{i+1}}{P_i}\right)^{N_i^e(t_0)} + \left(\frac{P_{i+1}^d}{P_i^d}\right)^{N_i^d(t_0)}, \quad (4.11)$$

where $\{P_i^d\}$ is given by equation (4.10). □

Here we provide a way to numerically approximate a lower bound of $Q_i(N_i^e(t_0), N_i^d(t_0))$. Consider a closed region in the (N_i^e, N_i^d) -plane defined by four line segments, which are divided into two groups $\mathcal{B}_r^L$ and $\mathcal{B}_c^L$.

$$\begin{aligned} \mathcal{B}_r^L : N_i^e &= 0, 0 \leq N_i^d \leq L, \text{ and } N_i^d = 0, 0 \leq N_i^e \leq L, \\ \mathcal{B}_c^L : N_i^e &= L, 0 < N_i^d \leq L, \text{ and } N_i^d = L, 0 < N_i^e \leq L, \end{aligned} \quad (4.12)$$

where $L > \max\{N_i^e(t_0), N_i^d(t_0)\}$. As shown in Fig. 4.10, $\mathcal{B}_r^L$ is represented by the red line and $\mathcal{B}_c^L$ is represented by the blue dashed line. The interior region is denoted by I .

Unlike the two-word dictionary, where the pair (N_i^e, N_i^d) behaves like a two-dimensional random walk with only 4 directions of movement, the pair (N_i^e, N_i^d) in a general dictionary can move in 8 directions or stand still. Denote the probability of going from (N_i^e, N_i^d) to $(N_i^e + x, N_i^d + y)$ by $P_{\{x,y\}}^i$, $x, y \in \{-1, 0, 1\}$. The transition probabilities of this random walk are shown in Table 4.6. Note that in the first four entries of the table, the summations actually include only one term. They have been written as summations to highlight the similar form of the various transition probabilities.

Table 4.6: Transition probability of a random walk in a general dictionary.

Direction	Transition Probability
$\{1, 1\}$	$P_{\{1,1\}}^i = \sum_{j=i} \rho^{d_{\{i,j\}}} (1 - \rho)^{m-d_{\{i,j\}}} P_i$
$\{-1, -1\}$	$P_{\{-1,-1\}}^i = \sum_{j=i+1} \rho^{d_{\{i+1,j\}}} (1 - \rho)^{m-d_{\{i+1,j\}}} P_{i+1}$
$\{1, -1\}$	$P_{\{1,-1\}}^i = \sum_{j=i+1} \rho^{d_{\{i,j\}}} (1 - \rho)^{m-d_{\{i,j\}}} P_i$
$\{-1, 1\}$	$P_{\{-1,1\}}^i = \sum_{j=i} \rho^{d_{\{i+1,j\}}} (1 - \rho)^{m-d_{\{i+1,j\}}} P_{i+1}$
$\{1, 0\}$	$P_{\{1,0\}}^i = \sum_{j \neq i, i+1} \rho^{d_{\{i,j\}}} (1 - \rho)^{m-d_{\{i,j\}}} P_i$
$\{-1, 0\}$	$P_{\{-1,0\}}^i = \sum_{j \neq i, i+1} \rho^{d_{\{i+1,j\}}} (1 - \rho)^{m-d_{\{i+1,j\}}} P_{i+1}$
$\{0, 1\}$	$P_{\{0,1\}}^i = \sum_{j \neq i, i+1} \rho^{d_{\{i,j\}}} (1 - \rho)^{m-d_{\{i,j\}}} P_j$
$\{0, -1\}$	$P_{\{0,-1\}}^i = \sum_{j \neq i, i+1} \rho^{d_{\{i+1,j\}}} (1 - \rho)^{m-d_{\{i+1,j\}}} P_j$
$\{0, 0\}$	$P_{\{0,0\}}^i = 1 - \sum_{x,y \in \{-1,0,1\}, \{x,y\} \neq \{0,0\}} P_{\{x,y\}}^i$

For any point (N^e, N^d) in the closed region, denote by $E_L(N^e, N^d)$ (or E_L for convenience) the event that a random walk starting at (N^e, N^d) will hit boundary group $\mathcal{B}_r^L$ before it hits $\mathcal{B}_c^L$ and denote by $\tilde{Q}_i(E_L)$ its probability. Clearly $E_{L_2} \subseteq E_{L_1}$ when $L_2 > L_1$, because if a random walk hits $\mathcal{B}_r^{L_1}$ before hitting $\mathcal{B}_c^{L_1}$, it also hits $\mathcal{B}_r^{L_2}$ before hitting $\mathcal{B}_c^{L_2}$. Therefore we have

$$\begin{aligned}
\tilde{Q}_i(E_L) &\leq \tilde{Q}_i(E_{L+1}) \leq \dots \leq Q_i, \\
\Rightarrow \tilde{Q}_i(N^e, N^d) &\stackrel{\text{def}}{=} \limsup_{L \rightarrow \infty} \tilde{Q}_i(E_L) \leq Q_i(N^e, N^d).
\end{aligned} \tag{4.13}$$

$\tilde{Q}_i(E_L(N^e, N^d))$ has the following properties.

$$\tilde{Q}_i(E_L(0, N^d)) = \tilde{Q}_i(E_L(N^e, 0)) = 1, \quad (4.14)$$

$$\tilde{Q}_i(E_L(L, N^d)) = 0 \text{ when } N^d > 0, \quad (4.15)$$

$$\tilde{Q}_i(E_L(N^e, L)) = 0 \text{ when } N^e > 0,$$

$$\tilde{Q}_i(E_L(N^e, N^d)) = \frac{\sum_{\{x,y\} \neq \{0,0\}} P_{\{x,y\}}^i \tilde{Q}_i(E_L(N^e + x, N^d + y))}{1 - P_{\{0,0\}}^i}. \quad (4.16)$$

To find $\tilde{Q}_i(E_L(N_i^e(t_0), N_i^d(t_0)))$, we first define the recurrence probability vector $\mathbf{Q}_i^L$ by

$$\mathbf{Q}_i^L = [\underbrace{\tilde{Q}_i(E_L(0,0)), \dots}_{\mathcal{B}_r^L}, \underbrace{\tilde{Q}_i(E_L(1,L)), \dots}_{\mathcal{B}_c^L}, \underbrace{\tilde{Q}_i(E_L(1,1)) \dots}_I]^\top \quad (4.17)$$

where, within each subvector, the ordering of the values corresponds to the lexicographical ordering of the locations $(N_i^e(t_0), N_i^d(t_0))$. Updating $\mathbf{Q}_i^L$ based on equations (4.14), (4.15) and (4.16) can then be represented by a matrix multiplication $\mathbf{T}\mathbf{Q}_i^L$. The transition matrix $\mathbf{T}$ has the form

$$\mathbf{T} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{R} & \mathbf{S} \end{bmatrix}. \quad (4.18)$$

It is easy to check that $\mathbf{T}$ satisfies the following properties.

- $\mathbf{T}$ is non-negative.
- The row sum $\sum_j t_{ij} = 1$ for all i .
- $\mathbf{I}$ is an identity matrix of size $4L$, representing the boundary conditions.
- $\mathbf{S}$ is irreducible and there exists row i such that $\sum_j s_{ij} < 1$.

- $\mathbf{T}^h$ can be represented as

$$\mathbf{T}^h = \left[\begin{array}{c|c} \mathbf{I} & \mathbf{0} \\ \hline \mathbf{R}(\mathbf{I} + \mathbf{S} + \dots + \mathbf{S}^{h-1}) & \mathbf{S}^h \end{array} \right]. \quad (4.19)$$

Since $\mathbf{S}$ has rows with row sum strictly less than 1, we can always find an irreducible matrix $\tilde{\mathbf{S}}$ such that $\tilde{\mathbf{S}} \geq \mathbf{S}$, $\tilde{\mathbf{S}} \neq \mathbf{S}$ and its row sum $\sum_j \tilde{s}_{ij} = 1$ for all rows. From [57, Lemma 1.2], the Perron root $\lambda(\mathbf{S})$ of $\mathbf{S}$ is strictly less than that of $\tilde{\mathbf{S}}$ and

$$\lambda(\mathbf{S}) < \lambda(\tilde{\mathbf{S}}) = 1. \quad (4.20)$$

Thus we have

$$\lim_{h \rightarrow \infty} \mathbf{S}^h = \mathbf{0}, \quad (4.21)$$

$$\lim_{h \rightarrow \infty} \mathbf{I} + \mathbf{S} + \dots + \mathbf{S}^{h-1} = (\mathbf{I} - \mathbf{S})^{-1}. \quad (4.22)$$

By combining equations (4.19), (4.21) and (4.22), we have

$$\lim_{h \rightarrow \infty} \mathbf{T}^h = \lim_{h \rightarrow \infty} \left[\begin{array}{c|c} \mathbf{I} & \mathbf{0} \\ \hline \mathbf{R}(\mathbf{I} + \mathbf{S} + \dots + \mathbf{S}^{h-1}) & \mathbf{S}^h \end{array} \right] = \left[\begin{array}{c|c} \mathbf{I} & \mathbf{0} \\ \hline \mathbf{R}(\mathbf{I} - \mathbf{S})^{-1} & \mathbf{0} \end{array} \right]. \quad (4.23)$$

Equation (4.23) indicates that $\tilde{Q}_i(E_L(N^e, N^d))$ is determined solely by the boundary values (equations (4.14) and (4.15)) and can be numerically approached by Algorithm 2. By increasing L , $\tilde{Q}_i(E_L(N^e, N^d))$ can be used to approach $\tilde{Q}_i(N^e, N^d)$, which is the lower bound on $Q_i(N^e, N^d)$.

Remark 4.3.1. This method can also be applied to the two-word dictionary. However, since the random walk moves in only four directions, as shown in Fig. 4.9, point (N^e, N^d) is only connected to other points $(\tilde{N}^e, \tilde{N}^d)$ when $N^e + N^d$ and $\tilde{N}^e + \tilde{N}^d$ are both even or odd. This implies that $\mathbf{S}$ is not irreducible. To solve this problem, we can further divide $\mathbf{S}$ into two irreducible components,

Algorithm 2 Approximating $\tilde{Q}_i(E_L(N_i^e(t_0), N_i^d(t_0)))$

Require: Area size L , index $(N_i^e(t_0), N_i^d(t_0))$, maximum iteration number J .

Ensure: Approximation of $\tilde{Q}_i(E_L(N_i^e(t_0), N_i^d(t_0)))$.

- 1: Initialize $\mathbf{Q}_i^L$ such that $\tilde{Q}_i(E_L(0, N^d)) = \tilde{Q}_i(E_L(N^e, 0)) = 1$ and $\tilde{Q}_i(E_L(N^e, L)) = \tilde{Q}_i(E_L(L, N^d)) = 0$. The remaining $\tilde{Q}_i(E_L N^e, N^d)$ are set to 0.
 - 2: **for** each iteration $j = 1, \dots, J$ **do**
 - 3: Update $\tilde{Q}_i(E_L(N^e, N^d))$ based on equations (4.14), (4.15) and (4.16).
 - 4: **end for**
-

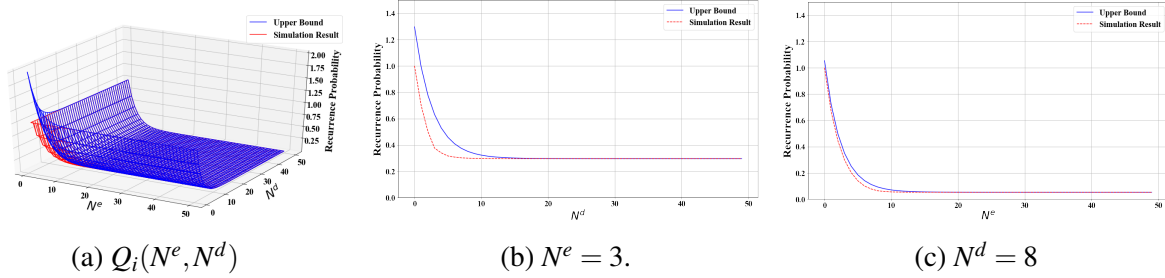


Figure 4.11: Recurrence probabilities of a two-word dictionary with distribution $P = \{0.6, 0.4\}$ and transition probability $\rho = 0.05$.

based on the parity of $N^e + N^d$. The transition matrix $\mathbf{T}$ can then be represented as

$$\mathbf{T} = \left[\begin{array}{c|cc} \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{R}_1 & \mathbf{S}_1 & \mathbf{0} \\ \mathbf{R}_2 & \mathbf{0} & \mathbf{S}_2 \end{array} \right] \quad (4.24)$$

and

$$\lim_{h \rightarrow \infty} \mathbf{T}^h = \left[\begin{array}{c|cc} \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{R}_1(\mathbf{I} - \mathbf{S}_1)^{-1} & \mathbf{0} & \mathbf{0} \\ \mathbf{R}_2(\mathbf{I} - \mathbf{S}_2)^{-1} & \mathbf{0} & \mathbf{0} \end{array} \right]. \quad (4.25)$$

In Fig. 4.11, the numerical results on the recurrence probabilities of a two-word dictionary with distribution $P = \{0.6, 0.4\}$ and transition probability $\rho = 0.05$ are presented. Both the upper bound in Theorem 4.3.2 and the simulation result using Algorithm 2 are shown. We can see that when N^e or N^d is large, the upper bound is tight.

4.3.3 Recurrence Probability of a General Dictionary

In this subsection we discuss the recurrence probability of the whole dictionary. Let W be the event that *a recurrence occurs involving any two adjacent words in a stable dictionary*, we have

$$W = \left(\bigcup_{i=1}^{2^{m-1}-1} \{i, i+1\}^e \right) \cup \left(\bigcup_{i=1}^{2^{m-1}-1} \{i, i+1\}^d \right). \quad (4.26)$$

Since a recurrence between words $\mathbf{w}_i$ and $\mathbf{w}_j$ always includes recurrence between adjacent words, we know that the probability of a recurrence involving any two words in the dictionary, denoted P_W , is

$$P_W = \Pr\{W\}. \quad (4.27)$$

Notice that $\{i, i+1\}^e$ and $\{i+1, i+2\}^e$ are not independent, but $\{i, i+1\}^e$ and $\{i+2, i+3\}^e$ are independent. This is also true for the decoding process. We now derive an upper bound on P_W . We have the following lemma.

Lemma 4.3.4. *The probability of recurrence P_W satisfies*

$$P_W \leq 4 - \prod_{i=1}^{2^{m-1}} \left(1 - \frac{P_{2i}}{P_{2i-1}} \frac{N_{2i-1}^e(t)}{N_{2i-1}^d(t)} \right) - \prod_{i=1}^{2^{m-1}-1} \left(1 - \frac{P_{2i+1}}{P_{2i}} \frac{N_{2i}^e(t)}{N_{2i}^d(t)} \right) - \prod_{i=1}^{2^{m-1}} \left(1 - \frac{P_{2i}^d}{P_{2i-1}^d} \frac{N_{2i-1}^d(t)}{N_{2i-1}^e(t)} \right) - \prod_{i=1}^{2^{m-1}-1} \left(1 - \frac{P_{2i+1}^d}{P_{2i}^d} \frac{N_{2i}^d(t)}{N_{2i}^e(t)} \right). \quad (4.28)$$

Proof. From the union bound, we have

$$P_W = \Pr\left\{ \left(\bigcup_{i=1}^{2^{m-1}-1} \{i, i+1\}^e \right) \cup \left(\bigcup_{i=1}^{2^{m-1}-1} \{i, i+1\}^d \right) \right\} \leq \Pr\left\{ \bigcup_{i=1}^{2^{m-1}-1} \{i, i+1\}^e \right\} + \Pr\left\{ \bigcup_{i=1}^{2^{m-1}-1} \{i, i+1\}^d \right\}. \quad (4.29)$$

Using the fact that $\{i, i+1\}^e$ and $\{i+2, i+3\}^e$ are independent, we have

$$\begin{aligned}
\Pr\left\{\bigcup_{i=1}^{2^m-1} \{i, i+1\}^e\right\} &= \Pr\left\{\left(\bigcup_{i=1}^{2^{m-1}} \{2i-1, 2i\}^e\right) \cup \left(\bigcup_{i=1}^{2^{m-1}-1} \{2i, 2i+1\}^e\right)\right\} \\
&\leq \Pr\left\{\bigcup_{i=1}^{2^{m-1}} \{2i-1, 2i\}^e\right\} + \Pr\left\{\bigcup_{i=1}^{2^{m-1}-1} \{2i, 2i+1\}^e\right\} \\
&= 2 - \Pr\left\{\left(\bigcup_{i=1}^{2^{m-1}} \{2i-1, 2i\}^e\right)^C\right\} - \Pr\left\{\left(\bigcup_{i=1}^{2^{m-1}-1} \{2i, 2i+1\}^e\right)^C\right\} \quad (4.30) \\
&= 2 - \Pr\left\{\left(\bigcap_{i=1}^{2^{m-1}} \overline{\{2i-1, 2i\}^e}\right)\right\} - \Pr\left\{\left(\bigcap_{i=1}^{2^{m-1}-1} \overline{\{2i, 2i+1\}^e}\right)\right\} \\
&= 2 - \prod_{i=1}^{2^{m-1}} \left(1 - \frac{P_{2i}}{P_{2i-1}}^{N_{2i-1}(t)}\right) - \prod_{i=1}^{2^{m-1}-1} \left(1 - \frac{P_{2i+1}}{P_{2i}}^{N_{2i}(t)}\right).
\end{aligned}$$

Similarly, for the decoding process we have

$$\Pr\left\{\bigcup_{i=1}^{2^m-1} \{i, i+1\}^d\right\} \leq 2 - \prod_{i=1}^{2^{m-1}} \left(1 - \frac{P_{2i}^d}{P_{2i-1}^d}^{N_{2i-1}(t)}\right) - \prod_{i=1}^{2^{m-1}-1} \left(1 - \frac{P_{2i+1}^d}{P_{2i}^d}^{N_{2i}(t)}\right). \quad (4.31)$$

Combining equations (4.29), (4.30), and (4.31) completes the proof. $\square$

Here we assume that the first t data words are encoded and decoded correctly, i.e. $\mathbf{n}^e(t) = \mathbf{n}^d(t)$. We denote by $A(\mathbf{n}^e(t))$ the right side of equation (4.28). Let $P(\mathbf{n}^e(t))$ denote the probability that after t steps the word counts are $\mathbf{n}^e(t)$. Then, we have

$$P(\mathbf{n}^e(t)) = \binom{t}{n_1^e(t), \dots, n_{2^m}^e(t)} P_1^{n_1^e(t)} \dots P_{2^m}^{n_{2^m}^e(t)}. \quad (4.32)$$

By combining Lemma 4.3.4 and the law of total probability, we have the following theorem.

Theorem 4.3.5. *After t data words are encoded and decoded correctly, the probability $P(t)$ that*

the dictionary will be unstable satisfies

$$P(t) \leq \sum_{\substack{\mathbf{n}^e(t) \\ \text{stable}}} A(\mathbf{n}^e(t))P(\mathbf{n}^e(t)) + \sum_{\substack{\mathbf{n}^e(t) \text{ not} \\ \text{stable}}} P(\mathbf{n}^e(t)). \quad (4.33)$$

□

The red dashed line in Fig. 4.12 shows the upper bound on $P(t)$ when we set $m = 2$, $P = \{0.4, 0.3, 0.2, 0.1\}$ and $\rho = 0.05, 0.1$ and 0.2 . Simulation results are also shown in the figure. The simulation consists of repeating the following steps 2,000 times.

- Generate a length-20,000 sequence with alphabet $\{11, 10, 01, 00\}$ (40,000 symbols from $\{0, 1\}$) and corresponding probability distribution $P = \{0.4, 0.3, 0.2, 0.1\}$.
- Apply SLC direct shaping code to the source sequence. After the first t words having been encoded, if a recurrence occurs during the encoding process, stop and declare a recurrence.
- Transmit the encoded sequence through a binary symmetric channel BSC(ρ) with transition probability ρ .
- Decode the received noisy sequence while making sure the first t symbols are decoded correctly (noise free).
- After the first t symbols having been decoded, if a recurrence occurs during the decoding process, stop and declare a recurrence.

The green line in Fig. 4.12 shows the fraction of experiments that stop because of recurrences. We see that $P(t)$ decreases rapidly as t increases. Other distributions and transition probabilities have also been simulated and they all produce qualitatively similar results. This indicates that if we make sure the first t data words are decoded correctly for large enough t , for example by combining shaping coding with error correction coding, we can significantly reduce the likelihood

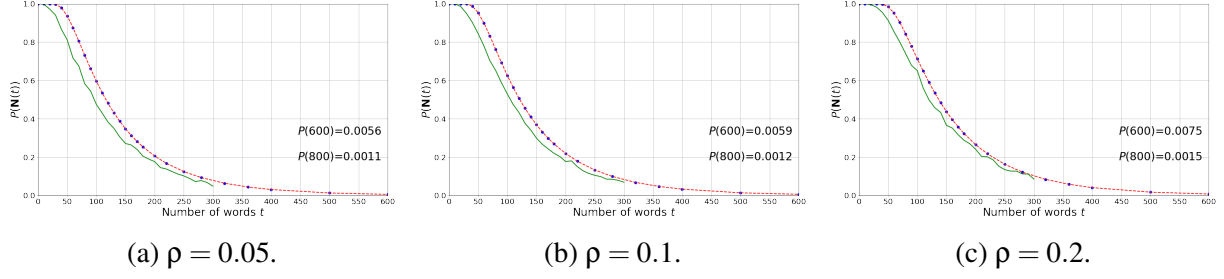


Figure 4.12: Upper bound and simulation result of $P(t)$ with different transition probability ρ .

of future error propagation. One possible approach is the bootstrap scheme in [10], which essentially uses a reverse concatenation architecture that combines systematic error correction codes and shaping codes, and has found application in optical data transmission.

Remark 4.3.2. $(\mathbf{n}^e(t), \mathbf{n}^d(t))$ acts like a random walk in a $(2^{m-1} - 1) \times (2^{m-1} - 1)$ space. Thus Algorithm 2 can be used to numerically approximate a lower bound on P_W . $\square$

4.3.4 Recurrence Behavior of *The Count of Monte Cristo*

Here we study the sequence generated by applying length-2 and length-4 direct shaping codes to TCMC. The fractions of input words for segments of TCMC are shown in Fig. 4.13. As shown in Fig. 4.13b and 4.13c, some words in the length-4 dictionary are not fully separated. This indicates that error propagation can happen throughout the entire encoding process.

We now examine the decoding recurrence probability of the length-2 direct shaping code. As shown in Fig. 4.13a, the last encoding recurrence of a length-2 direct shaping code happens between words 00 and 11 at $\gamma = 40,346$ bits. We did the following simulation, which consists of repeating the following steps 2000 times.

- Encode the first 10,000 bytes of TCMC (80,000 bits, or 40,000 input words).
- Transmit the encoded sequence through a binary symmetric channel BSC(ρ) with transition probability ρ .

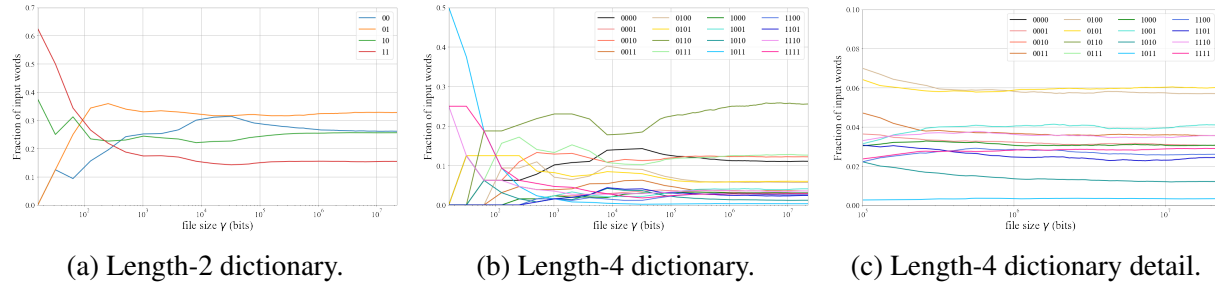


Figure 4.13: Fraction of words for segments of *The Count of Monte Cristo*.

- Decode the received noisy sequence while making sure the first t symbols are decoded correctly (noise free).
- After the first t symbols having been decoded, if a recurrence occurs during the decoding process, stop and declare a recurrence.

The probability of decoding recurrence $P_d(t)$ with different ρ is shown in Fig. 4.14. As shown in the figure, $P_d(t)$ quickly converges to 0 when $t \geq 40,346$ and the rate of convergence increases as ρ decreases.

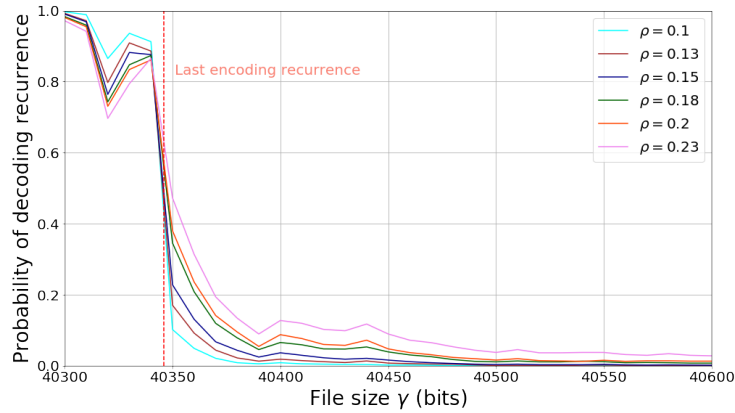


Figure 4.14: Probability of decoding recurrence when segments of *The Count of Monte Cristo* are decoded correctly.

4.4 Performance of Direct Shaping Codes

The performance of rate-constrained shaping codes was studied in Chapter 3. Given an i.i.d. source $\mathbf{X} = X_1, X_2, \dots$ with entropy H and a variable-length mapping $\phi : \mathcal{X}^q \rightarrow \mathcal{Y}^*$ used as a shaping code, the relationship between the code expansion factor f , which is the ratio of the expected codeword length to the input word length, and the average cost was analyzed. In particular, the minimum achievable average cost associated with a fixed expansion factor was determined.

In Remark 3.2.1, it was shown that direct shaping codes are shaping codes with parameters $q = 1$ and $f = 1$, where both the input and output processes have alphabet size 2^m . The cost process $\mathbf{C} = \{C_t\}$ of a direct shaping code ϕ is

$$C_1 = c(\phi(X_1)), C_2 = \frac{c(\phi(X_1, X_2))}{2}, \dots, C_t = \frac{c(\phi(X_1, X_2, \dots, X_t))}{t}, \quad (4.34)$$

and the asymptotic expected average cost is defined as $C_\infty = \lim_{t \rightarrow \infty} E(C_t)$. In this section, we discuss the performance of direct shaping codes under the assumption of no errors. The following theorem indicates the asymptotic expected average cost achievable by a direct shaping code.

Theorem 4.4.1. *Given the distribution P and cost vector C , the asymptotic expected average cost C_∞ of a direct shaping code is $\sum_i P_i c_i / m$.*

Proof. We first assume that $P_1 > P_2 > \dots > P_{2^m}$. Consider a sequence of i.i.d. random variables $X_1^{ij}, X_2^{ij}, \dots$ with $P(X_1^{ij} = 1) = P_i$, $P(X_1^{ij} = -1) = P_j$ and $P(X_1^{ij} = 0) = 1 - P_i - P_j$. The random variable X_k^{ij} corresponds to the change in distance between symbol i and j at time k . The expected value of X_1^{ij} is $\mu^{ij} = P_i - P_j$. Define random variable $S_t^{ij} = \sum_{k=1}^t X_k^{ij}$ and note that $S_t^{ij} > 0$ means $n_i(t) > n_j(t)$. By the strong law of large numbers,

$$S_t^{ij} / t - \mu^{ij} \rightarrow 0 \quad \text{a.s.} \quad (4.35)$$

For any $\varepsilon > 0$, $\{S_t^{ij}/t\}_0^\infty$ is within ε of $\mu^{ij} = P_i - P_j > 0$ for all but finitely many t . This means for any two symbols $i > j$, $n_i(t) > n_j(t)$ for all but finitely many t . In other words for any $\varepsilon, \delta > 0$, there exists T_1 such that the probability of event $n_i(t) - n_j(t) > \varepsilon$ for any $P_i > P_j$ and $t > T_1$ is greater than $1 - \delta$. This indicates the dictionary is stable almost surely, i.e., after some time T , with probability greater than $1 - \delta$, the dictionary is stable and a recurrence never occurs.

Here we define another i.i.d. process $\{\tilde{c}(X_t)\}$ with $\tilde{c}(X) = c_i$ if $X = \mathbf{w}_i$. By the strong law of large numbers, we know that

$$\tilde{C}_t = \frac{\tilde{c}(X_1, X_2, \dots, X_t)}{t} = \frac{\sum_i \tilde{c}(X_i)}{t} \rightarrow \sum P_i c_i \quad \text{a.s.} \quad (4.36)$$

where we implicitly extend the definition of $\tilde{c}$ to vectors additively. If after symbols X_1^{t-1} the dictionary is stable, for the next input X_t we have

$$\tilde{c}(X_t) = c(\phi(X_1, X_2, \dots, X_t)) - c(\phi(X_1, X_2, \dots, X_{t-1})). \quad (4.37)$$

Since the dictionary is stable almost surely, for any $\delta, \varepsilon > 0$, there exists T_1 such that with probability greater than $1 - \delta/2$, the dictionary is stable for any $t > T_1$ and there exists T_2 such that with probability greater than $1 - \delta/2$, $|\tilde{C}_t - \sum P_i c_i| < \frac{\varepsilon}{2}$ for all $t > T_2$. Therefore with probability greater than $1 - \delta$, for any $t > T' = \max\{T_1, T_2\}$, we have

$$\begin{aligned} C_t &= \frac{c(\phi(X_1, X_2, \dots, X_t))}{t} \\ &= \frac{c(\phi(X_1, X_2, \dots, X_{T'})) + c(\phi(X_1, X_2, \dots, X_t)) - c(\phi(X_1, X_2, \dots, X_{T'}))}{t} \\ &= \frac{c(\phi(X_1, X_2, \dots, X_{T'})) + \tilde{c}(X_{T'+1} \dots X_t)}{t} \\ &= \frac{c(\phi(X_1, X_2, \dots, X_{T'})) - \tilde{c}(X_1, X_2, \dots, X_{T'}) + \tilde{c}(X_1, X_2, \dots, X_t)}{t}, \end{aligned} \quad (4.38)$$

and

$$\left| \frac{\tilde{c}(X_1, X_2, \dots, X_t)}{t} - \sum P_i c_i \right| < \frac{\varepsilon}{2}. \quad (4.39)$$

Since the cost is bounded by c_1 and c_{2^m} , for the first T' steps we have

$$|c(\phi(X_1, X_2, \dots, X_{T'})) - \tilde{c}(X_1, X_2, \dots, X_{T'})| \leq T'(c_1 - c_{2^m}). \quad (4.40)$$

Combining equations (4.38), (4.39) and (4.40), we have

$$\begin{aligned} |C_t - \sum P_i c_i| &\leq |C_t - \tilde{C}_t| + |\tilde{C}_t - \sum P_i c_i| \\ &< \frac{|c(X_1, X_2, \dots, X_{T'}) - \tilde{c}(X_1, X_2, \dots, X_{T'})|}{t} + \frac{\epsilon}{2} \\ &\leq (c_1 - c_{2^m}) \frac{T'}{t} + \frac{\epsilon}{2}. \end{aligned} \quad (4.41)$$

Let $T = \max\{T_1, T_2, \frac{2}{\epsilon}(c_1 - c_{2^m})T'\}$, then with probability greater than $1 - \delta$, $|C_t - \sum P_i c_i| < \epsilon$ for any $t > T$. This indicates that

$$C_t \rightarrow \sum P_i c_i \quad \text{a.s..} \quad (4.42)$$

Since C_t is bounded by c_1 and c_{2^m} , by the dominated convergence theorem we have

$$C_\infty = \lim_{t \rightarrow \infty} E(C_t) = \sum P_i c_i. \quad (4.43)$$

When there exists some i such that $P_i = P_{i+1}$, equation (4.35) becomes

$$S_t^{ii+1}/t - 0 \rightarrow 0 \quad \text{a.s..} \quad (4.44)$$

This means that the almost sure stability condition is not satisfied. However, this doesn't affect the conclusion, because almost surely the input words $\mathbf{w}_i$ and $\mathbf{w}_{i+1}$ are mapped to $\mathbf{y}_i$ and $\mathbf{y}_{i+1}$ and $P(\mathbf{y}_i) = P(\mathbf{y}_{i+1}) = P_i$. This implies that the expected value of average cost satisfies $C_\infty = \lim_{t \rightarrow \infty} E(C_t) = \sum P_i c_i$. This completes the proof. $\square$

It was proved in Theorem 3.3.1 that the minimum average cost of a rate-1 shaping code

for costly channels with cost vector C and an i.i.d. source with entropy H is $\sum_i \hat{P}_i c_i$, where

$$\hat{P}_i = \frac{2^{-\mu c_i}}{\sum_j 2^{-\mu c_j}}, \quad \mu \geq 0 \text{ and } -\sum_i \hat{P}_i \log_2 \hat{P}_i = H. \quad (4.45)$$

This indicates that direct shaping codes are in general suboptimal. We have the following corollary.

Corollary 4.4.1.1. *For an i.i.d. source with entropy H , SLC direct shaping codes are optimal with respect to average wear cost if and only if*

$$P_i = \frac{2^{-\mu c_i}}{\sum_j 2^{-\mu c_j}}, \quad \mu \geq 0 \text{ and } -\sum_i P_i \log_2 P_i = H. \quad (4.46)$$

4.5 Conclusion

In this paper, we studied shaping codes to reduce programming wear when writing structured data to flash memory. We first reviewed so-called direct shaping codes for SLC flash memory. Using a page-oriented, programming cost model, we then extended this technique to MLC flash memory. The performance of these shaping codes was empirically evaluated on English and Chinese language text. Then we examined the error propagation behavior of direct shaping codes. We showed that by making sure the first t data words are decoded correctly for large enough t , we can significantly reduce the likelihood of future error propagation. Finally, we derived the asymptotic average cost of a direct shaping code. The results indicate that direct shaping codes are in general suboptimal and can only achieve the minimum average cost for rate-1 codes if the source distribution satisfies a specific condition.

4.6 Acknowledgment

Chapter 4 is in part a reprint of the material in the paper: Yi Liu and Paul H. Siegel, On the Performance of Direct Shaping Codes, which has been submitted to *IEEE Transactions on Communications*. It is also in part a reprint of materials in the papers: Yi Liu and Paul H. Siegel, “Shaping Codes for Structured Data,” in *Proc. IEEE Globecom*, Washington, D.C., Dec. 4-8, 2016, pp. 1–5, and Yi Liu and Paul H. Siegel, “Shaping Codes for Structured Data,” in *Proc. 7th Annu. Non-Volatile Memories Workshop (NVMW)*, La Jolla, CA, Mar. 6-8, 2016. The dissertation author was the primary investigator and author of these papers.

Chapter 5

Rate-Constrained Shaping Codes for Noiseless Finite-State Costly Channels

5.1 Capacity of the Noiseless Finite-State Costly Channel

In this section, we introduce the noiseless finite-state costly channel and background results on its capacity. Let $\mathcal{H} = (\mathcal{S}, \mathcal{U})$ be a finite, directed graph, with state set $\mathcal{S}$, and edge set $\mathcal{U}$, which is also irreducible. Denote by $s_i \in \mathcal{S}$ the state in $\mathcal{H}$. A *finite-state costly channel* is a noiseless channel associated with $\mathcal{H}$, where each edge $u \in \mathcal{U}$ is assigned a non-negative cost $w(u) \geq 0$. We assume that between any states pair $(s_i, s_j) \in \mathcal{S} \times \mathcal{S}$, there is at most one edge. If not, we can always convert it to another graph that satisfies this condition by using edge and state splitting. This requirement implies that each edge can be fully defined by its initial and ending state, denoted by u_{ij} . An example of such channel is given in Example 5.1.1.

Example 5.1.1. In Chapter 3, NAND flash memory is modeled as a memoryless costly channel. Since the cells are arranged in a block, programming a cell affects its neighbors. One example of this phenomenon is *inter-cell inference* (ICI). In [72], the encoding errors caused by ICI were characterized and constrained codes were used to mitigate ICI-susceptible data patterns. In this

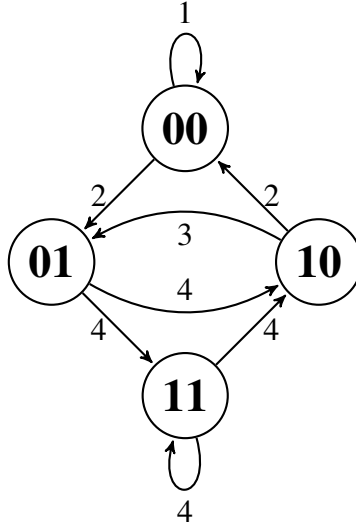


Figure 5.1: Flash memory channel

chapter, we consider single-level cell (SLC) flash memory. In SLC flash memory, cells have two states: programmed, corresponding to bit 1, or erased, corresponding to bit 0. Due to ICI, programming a cell will damage its neighbors cells. Here we assume that each length-3 sequence has the cost shown in Table 5.1. We can convert this table into a directed graph with state set

Table 5.1: Flash memory channel Cost

u	000	001	010	011	100	101	110	111
$C(u)$	1	2	4	4	2	3	4	4

$\mathcal{S} = \{00, 01, 10, 11\}$, as shown in Fig. 5.1.

5.1.1 Channel Capacity with Average Cost Constraint

Given a length- n edge sequence u_1^n , the cost of this sequence is defined as

$$C(u_1^n) = \sum_{i=1}^n w(u_i), \quad (5.1)$$

and the average cost of this sequence is defined as

$$A(u_1^n) = \frac{1}{n}C(u_1^n). \quad (5.2)$$

If we denote by $K_n(W)$ the number of sequences of length- n with average cost less than or equal to W , then the *combinatorial capacity* for a given average cost constraint is [70]

$$C_{I,\text{comb}}(W) \stackrel{\text{def}}{=} \limsup_{n \rightarrow \infty} \frac{1}{n} \log_2 |K_n(W)|. \quad (5.3)$$

We also refer to this definition as *type-I combinatorial capacity*. Let $\mathbf{U}$ be a stationary Markov process with entropy rate $H(\mathbf{U})$ and average cost $A(\mathbf{U})$. The *probabilistic capacity* for a given average cost constraint W is

$$C_{I,\text{prob}}(W) \stackrel{\text{def}}{=} \sup_{\mathbf{U}: A(\mathbf{U}) \leq W} H(\mathbf{U}). \quad (5.4)$$

The maxentropic Markov chain for a given W was derived in [32] and [52]. The result relies on the one-step cost-enumerator matrix $D(B)$, where $B \geq 0$, with entries

$$d_{ij}(B) = \begin{cases} 2^{-Bw(u_{ij})} \\ 0 & \text{if edge } u_{ij} \text{ doesn't exist.} \end{cases} \quad (5.5)$$

Denote by $\lambda(B)$ its Perron root and by vectors $E_L = [P_{s_i}/v_i]$ and $E_R = [v_i]^\top$ the corresponding left and right eigenvectors such that $E_L E_R = 1$. Given an average cost constraint $W(B)$ the maxentropic Markov chain has transition probabilities

$$P_{ij}(B) = \frac{1}{v_i} \frac{2^{-Bw(u_{ij})}}{\lambda(B)} v_j, \quad (5.6)$$

such that

$$W(B) = \frac{1}{\lambda(B)} \sum_{ij} P_{s_i} 2^{-Bw(u_{ij})} \frac{v_j}{v_i}, \quad (5.7)$$

and the type-I probabilistic capacity of this channel is

$$C_{I,\text{prob}}(W(B)) = \log_2 \lambda(B) + BW(B). \quad (5.8)$$

It was shown in [70] that $C_{I,\text{comb}}(W) = C_{I,\text{prob}}(W)$.

5.1.2 Channel Capacity without Cost Constraint

Assuming that all edge costs are integers, denote by $K(W)$ the number of distinct sequences u^* with costs equal to W . The combinatorial capacity, or the *type-II combinatorial capacity* of this channel is defined as

$$C_{II,\text{comb}} \stackrel{\text{def}}{=} \limsup_{W \rightarrow \infty} \frac{1}{W} \log_2 K(W). \quad (5.9)$$

Similarly, the *type-II probabilistic capacity* of this channel is define as

$$C_{II,\text{prob}} \stackrel{\text{def}}{=} \sup_{\mathbf{U}} \frac{H(\mathbf{U})}{A(\mathbf{U})}. \quad (5.10)$$

In [35], it was proved that the transition probability of the maxentropic Markov process is $P_{ij}(B_0)$ in equation (5.6), where B_0 satisfies $\lambda(B_0) = 1$. It was also proved that

$$C_{II,\text{comb}} = C_{II,\text{prob}} = B_0. \quad (5.11)$$

In [13], the author extended the definition of combinatorial capacity to channels with arbitrary non-negative costs. In [5] and [9], the equivalence between $C_{II,\text{comb}}$ and $C_{II,\text{prob}}$ when the graph is dense was proved.

5.2 Finite-State Variable-Length Codes: A Word-Valued Source Approach

5.2.1 Finite-State Variable-Length Codes

Let $\mathbf{X} = X_1 X_2 \dots$, where $X_i \sim X$ for all i , be an i.i.d. source with alphabet $\mathcal{X} = \{\alpha_1, \dots, \alpha_u\}$. We denote by P_i the probability of symbol α_i and without loss of generality we assume that $P_1 \geq P_2 \geq \dots \geq P_u$. We use $|\mathcal{X}|$ to denote the size of the alphabet and use $P(x^*)$ to denote the probability of any finite sequence x^* . A finite-state variable-length code on graph $\mathcal{H}$ is a mapping $\phi : S \times \mathcal{X}^q \rightarrow U^*$. For simplicity and without loss of generality, here we assume $q = 1$ and we denote $\phi(s_i, x_j)$ as $y_{\{i,j\}}$. The starting state of $y_{\{i,j\}}$ is s_i and is denoted by $start(y_{\{i,j\}})$. Its ending state is denoted by $end(y_{\{i,j\}})$. Its length is denoted by $L(y_{\{i,j\}})$. Throughout this chapter, we assume this mapping has the following two properties.

- Its subcodebook set $\mathcal{Y}_i = \{y_{\{i,j\}}\}$ is prefix-free for all states s_i .
- $l \leq L(y_{\{i,j\}}) < L$ for all s_i and x_j , where l and L are constants.

The encoding process associated with this mapping, as shown in Fig. 5.2, is defined as follows.

- Start: fixed state s_0 .
- Input words: sequence $x_1, x_2, x_3, \dots$
- Output words: codeword sequence $y_1, y_2, y_3, \dots \in \mathcal{U}^*$.
- Encoding rules:

$$\begin{aligned} y_i &= \phi(s_{i-1}, x_i), \\ s_i &= end(y_i). \end{aligned} \tag{5.12}$$

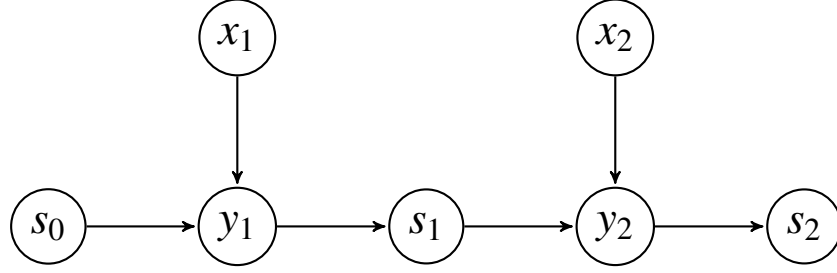


Figure 5.2: Encoding process.

From the encoding rules (5.12), we have

$$\begin{aligned} y_i &= \phi(y_{i-1}, x_i) = \phi(\text{end}(y_{i-1}), x_i) = F(y_{i-1}, x_i), \\ s_i &= \text{end}(y_i) = \text{end}(\phi(s_{i-1}, x_i)) = G(s_{i-1}, x_i). \end{aligned} \tag{5.13}$$

These equations suggest that we can define *codeword graph* $\mathcal{F}_0$ and *state graph* $\mathcal{G}_0$, which are closely related to the encoding process, as follows.

- Codeword graph $\mathcal{F}_0 = (V_{F_0}, E_{F_0})$:
 - Vertex set $V_{F_0} = \{y_{\{i,j\}} \text{ for all } i \text{ and } j\}$.
 - Edge set $E_{F_0} = \{e = (y_{\{i,j\}}, y_{\{k,l\}}) | s_k = \text{end}(y_{\{i,j\}})\}$.
- State graph $\mathcal{G}_0 = (V_{G_0}, E_{G_0})$:
 - Vertex set $V_{G_0} = \mathcal{S}$.
 - Edge set $E_{G_0} = \{e = (s_i, s_j) | \exists x_k \text{ s.t. } s_j = G(s_i, x_k)\}$.

We already assume that $\mathcal{H}$ is irreducible, but this does not guarantee that $\mathcal{F}_0$ and $\mathcal{G}_0$ are also irreducible. Here we provide a way to generate an irreducible subgraph from $\mathcal{F}_0$ and $\mathcal{G}_0$.

We choose the irreducible component $\mathcal{G} \subseteq \mathcal{G}_0$ that contains s_0 (we assume that s_0 always belongs to one of the irreducible components). Its state and edge sets are denoted by V_G and E_G respectively. For convenience, we will re-index the elements in V_G as $s_0, s_1, s_2, \dots$. The encoding

process is a Markov process associated with $\mathcal{G}$ and transition probabilities

$$t_{ij}^G = \sum_{k: G(s_i, x_k) = s_j} P(x_k). \quad (5.14)$$

We denote by $\pi_G(s_i)$ the stationary distribution of state s_i . Subgraph $\mathcal{F} \subset \mathcal{F}_0$ is built based on the following rules. Its state set V_F is

$$V_F = \{y_{\{i,j\}} = \phi(s_i, x_j) \mid \text{for } s_i \in V_G \text{ and all } x_j\} \quad (5.15)$$

and its edge set E_F is

$$E_F = \{(y_{\{i,j\}}, y_{\{k,l\}}) \mid y_{\{i,j\}}, y_{\{k,l\}} \in V_F \text{ and } s_k = \text{end}(y_{\{i,j\}}) = \text{end}(\phi(s_i, x_j))\}. \quad (5.16)$$

We can prove the following lemma, which shows that $\mathcal{F}$ is irreducible.

Lemma 5.2.1. *The graph $\mathcal{F}$ is irreducible. The encoding process is a Markov process associated with $\mathcal{F}$ with transition probabilities*

$$P(y_{\{k,l\}} \mid y_{\{i,j\}}) = P(x_l). \quad (5.17)$$

The stationary distribution of states $\{y_{\{i,j\}}\}$ is

$$\pi_F(y_{\{i,j\}}) = \pi_G(s_i)P(x_j). \quad (5.18)$$

Proof. We first show that $\mathcal{F}$ is irreducible. For any pair of vertices $(y_{\{i,j\}}, y_{\{k,l\}})$ in $\mathcal{F}$, by definition the ending state of $y_{\{i,j\}}$ is $s_m = \text{end}(y_{\{i,j\}})$ and the starting state of $y_{\{k,l\}}$ is s_k . Both states are contained in $\mathcal{G}$, which means there exists a path from s_m to s_k . This implies that there exists paths from $y_{\{i,j\}}$ to $y_{\{k,l\}}$ and therefore $\mathcal{F}$ is irreducible.

We now calculate the stationary distribution of the encoding process associated with $\mathcal{F}$.

The entries of the transition matrix $T_F = \{t_{y_{\{i,j\}}, y_{\{k,l\}}}^F\}$ satisfy

$$t_{y_{\{i,j\}}, y_{\{k,l\}}}^F = \begin{cases} P(x_l) & \text{if } \text{end}(y_{\{i,j\}}) = s_k \\ 0 & \text{elsewhere.} \end{cases} \quad (5.19)$$

Consider the vector

$$E_L = [\pi_G(s_1)P(x_1), \pi_G(s_1)P(x_2), \dots, \pi_G(s_1)P(x_u), \\ \pi_G(s_2)P(x_1) \dots, \pi_G(s_i)P(x_j), \dots, \pi_G(s_n)P(x_u)]. \quad (5.20)$$

It is easy to check that the sum of entries in E_L is 1. Since $\{\pi_G(s_i)\}$ is the stationary distribution of graph $\mathcal{G}$, we have

$$\sum_{i,j: \text{end}(y_{\{i,j\}}) = s_k} \pi_G(s_i)P(x_j) = \pi_G(s_k), \quad (5.21)$$

and

$$\sum_{i,j: \text{end}(y_{\{i,j\}}) = s_k} P(x_l) \pi_G(s_i)P(x_j) = \pi_G(s_k)P(x_l) \quad (5.22)$$

for any x_l . By combining equations (5.19), (5.20) and (5.22), we have

$$E_L T_F = E_L. \quad (5.23)$$

This implies that the stationary distribution of the encoding process is

$$\pi_F(y_{\{i,j\}}) = \pi_G(s_i)P(x_j). \quad (5.24)$$

This finishes the proof. □

5.2.2 Asymptotic Performance

In this subsection, we study the codeword process $\mathbf{U}$. The following theorem, stated in [15, Exercise 5.5], shows the asymptotic performance of the encoding process associated with graph $\mathcal{F}$.

Theorem 5.2.2 (Law of large numbers for irreducible Markov chain). *For an irreducible Markov chain with stationary distribution π_F . Let A be a function that satisfies*

$$\sum_{y_{\{i,j\}} \in V_F} |A(y_{\{i,j\}})| \pi_F(y_{\{i,j\}}) < \infty. \quad (5.25)$$

Then we have

$$\frac{1}{n} \sum_{m=1}^n A(Y_m) \rightarrow \sum_{y_{\{i,j\}} \in V_F} A(y_{\{i,j\}}) \pi_F(y_{\{i,j\}}) \quad P_\mu - a.s. \quad (5.26)$$

for any initial distribution μ , where $Y_1 Y_2 \dots Y_n$ is the codeword process. $\square$

Since $L(y_{\{i,j\}}) < \mathcal{L}$, from Theorem 5.2.2 we have

$$\frac{1}{n} \sum_{m=1}^n L(Y_m) \rightarrow \sum_{ij} L(y_{\{i,j\}}) \pi_F(y_{\{i,j\}}) = \sum_{ij} L(y_{\{i,j\}}) \pi_G(s_i) P(x_j) \quad P_\mu - a.s.. \quad (5.27)$$

By the dominated convergence theorem [15, p.15], we have

$$\mathbb{E}(L) \stackrel{\text{def}}{=} \lim_{n \rightarrow \infty} \frac{1}{n} \mathbb{E} \left(\sum_{m=1}^n L(Y_m) \right) = \sum_{ij} L(y_{\{i,j\}}) \pi_G(s_i) P(x_j). \quad (5.28)$$

Given any edge $u \in \mathcal{U}$ and codeword $y_{\{i,j\}}$, we denote by $N_u(y_{\{i,j\}})$ the total number of occurrences of u in sequence $y_{\{i,j\}}$, i.e.,

$$N_u(y_{\{i,j\}}) = \sum_{k=1}^{L(y_{\{i,j\}})} \mathbb{1}_{[y_{\{i,j\}}]_k^{k+1}=u}. \quad (5.29)$$

Similar to equations (5.27) and (5.28), from Theorem 5.2.2 and the dominated convergence

theorem, we have

$$\frac{1}{n} \sum_{m=1}^n N_u(Y_m) \rightarrow \sum_{ij} N_u(y_{\{i,j\}}) \pi_F(y_{\{i,j\}}) = \sum_{ij} N_u(y_{\{i,j\}}) \pi_G(s_i) P(x_j) \quad P_\mu - \text{a.s.}, \quad (5.30)$$

and

$$\mathbb{E}(N_u) \stackrel{\text{def}}{=} \lim_{n \rightarrow \infty} \frac{1}{n} \mathbb{E} \left(\sum_{m=1}^n N_u(Y_m) \right) = \sum_{y_{\{i,j\}}} N_u(y_{\{i,j\}}) \pi_G(s_i) P(x_j). \quad (5.31)$$

5.2.3 Finite-State Word-Valued Source

Introduced in [56], a *word-valued source* is a discrete random process that is formed by sequentially encoding the symbols of an i.i.d. random process $\mathbf{X}$ with codewords from a one-to-one mapping. This work was extended to an ergodic and stationary process in [22], and further to an asymptotically mean stationary process in [73]. In this chapter, the mapping ϕ is a function of both input symbols and current states. Therefore we refer to the process $\mathbf{U}$, which is formed by an i.i.d. process and ϕ , as the *finite-state word-valued source*. In this subsection, we prove theorems regarding the properties of the finite-state word-valued source.

Given an encoded sequence $U_1 U_2 \dots$ from mapping ϕ , the probability of sequence u_1^n is denoted by $Q(u_1^n)$. We are interested in the following limits.

- The asymptotic symbol occurrence probability

$$\hat{P}_u = \lim_{n \rightarrow \infty} \frac{1}{n} \mathbb{E}(N_u(U_1^n)). \quad (5.32)$$

- The asymptotic average cost

$$A(\phi) = \lim_{n \rightarrow \infty} \frac{1}{n} \mathbb{E}(C(u_1^n)). \quad (5.33)$$

- The entropy rate

$$H(\mathbf{U}) = \lim_{n \rightarrow \infty} \frac{1}{n} H(U_1^n). \quad (5.34)$$

In the rest of this subsection, we will prove that these limits exist and calculate their values.

Define the prefix operator γ as $u_1^n \gamma^i = z_1^{n-i}$ for $0 \leq i < n$ and $u_1^n \gamma^i = \lambda$ for $i \geq n$. Let $\gamma\{z^*\}$ denote the set of all the prefixes of a sequence z^* . We denote by $\mathcal{G}_\phi(u_1^n)$ the set of all sequences $x^* \in \mathcal{X}^*$ such that u_1^n is a prefix of $\phi(x^*)$ but not of $\phi(x^* \gamma)$. That is,

$$\mathcal{G}_\phi(u_1^n) = \{x^* \in \mathcal{X}^* \mid u_1^n \in \gamma\{\phi(x^*)\} \wedge |\phi(x^* \gamma)| < n\} \quad (5.35)$$

and the distribution of u_1^n can be expressed as

$$Q(u_1^n) = \sum_{x^* \in \mathcal{G}_\phi(y_1^n)} P(x^*). \quad (5.36)$$

We define by M_n the minimum length of a sequence x_1^m such that $|\phi(x_1^m)| \geq n$. Note that

$$L_{Y_1^{M_n-1}} < n \leq L_{Y_1^{M_n}}. \quad (5.37)$$

M_n is a stopping rule. Since $L(y_{\{i,j\}})$ is upper bounded, we have

$$\lim_{n \rightarrow \infty} M_n = \infty. \quad (5.38)$$

We can prove the following lemma.

Lemma 5.2.3. *For a finite-state variable-length code $\phi : \mathcal{S} \times \mathcal{X}^q \rightarrow \mathcal{U}^*$ associated with graph $\mathcal{F}$ and stationary distribution $\{\pi_F y_{\{i,j\}} = \pi_G(s_i) P(x_j)\}$ such that $\mathbb{E}(N_u) < \infty$ for all $u \in \mathcal{U}$ and*

$\mathbb{E}(L) < \infty$, the asymptotic probability of occurrence $\hat{P}_u$ is given by

$$\hat{P}_u = \mathbb{E}(N_u) \frac{1}{\mathbb{E}(L)}. \quad (5.39)$$

□

Proof. From equations (5.27) and (5.30), we have

$$\lim_{n \rightarrow \infty} \frac{1}{M_n} L(Y_1^{M_n}) = \mathbb{E}(L) \quad \text{a.s.}, \quad (5.40)$$

$$\lim_{n \rightarrow \infty} \frac{1}{M_n} N_u(Y_1^{M_n}) = E(N_r) \quad \text{a.s.} \quad (5.41)$$

By combining equations (5.38) and (5.40), we have

$$\frac{n}{M_n} \leq \frac{L(Y_1^{M_n})}{M_n} \rightarrow \mathbb{E}(L) \quad \text{a.s.}, \quad (5.42)$$

$$\frac{n}{M_n} > \frac{L(Y_1^{M_n-1})}{M_n} = \frac{L(Y_1^{M_n-1})}{M_n-1} \frac{M_n-1}{M_n} \rightarrow \mathbb{E}(L) \quad \text{a.s.} \quad (5.43)$$

This implies that

$$\frac{n}{M_n} \rightarrow \mathbb{E}(L) \quad \text{a.s.} \quad (5.44)$$

Similarly, by combining equations (5.38) and (5.41), we have

$$\frac{N_u(U_1^n)}{M_n} \rightarrow \mathbb{E}(N_u) \quad \text{a.s. for all } u \in \mathcal{U}. \quad (5.45)$$

From equations (5.44) and (5.45), we have

$$\lim_{n \rightarrow \infty} \frac{1}{n} N_u(U_1^n) = \frac{\mathbb{E}(N)}{\mathbb{E}(L)} \quad \text{a.s.}, \quad (5.46)$$

and the dominated convergence theorem implies that

$$\lim_{n \rightarrow \infty} \frac{1}{n} \mathbb{E}(N_u(U_1^n)) = \frac{\mathbb{E}(N_u)}{\mathbb{E}(L)}. \quad (5.47)$$

This finishes the proof. $\square$

Using Lemma 5.2.3, we can derive the asymptotic average cost $A(\phi)$. We have

$$\begin{aligned} A(\phi) &= \lim_{n \rightarrow \infty} \frac{1}{n} \mathbb{E}(C(u_1^n)) \\ &= \lim_{n \rightarrow \infty} \frac{\sum_{u_1^n \in \mathcal{U}^n} Q(u_1^n) C(u_1^n)}{n} \\ &= \lim_{n \rightarrow \infty} \frac{\sum_{u_1^n \in \mathcal{U}^n} Q(u_1^n) \sum_{u \in \mathcal{U}} N_u(u_1^n) w(u)}{n} \\ &= \sum_{u \in \mathcal{U}} w(u) \lim_{n \rightarrow \infty} \frac{\sum_{u_1^n \in \mathcal{U}^n} Q(u_1^n) N_u(u_1^n)}{n} \\ &= \sum_{u \in \mathcal{U}} \hat{P}_u w(u). \end{aligned} \quad (5.48)$$

The following lemma gives the entropy rate of a finite-state word valued source.

Lemma 5.2.4. *For a finite-state variable-length code $\phi : \mathcal{S} \times X^q \rightarrow \mathcal{U}^*$ associated with graph $\mathcal{F}$ such that $H(\mathbf{X}) < \infty$ and $\mathbb{E}(L) < \infty$, the entropy rate of the codeword process is*

$$H(\mathbf{U}) = \frac{qH(\mathbf{X})}{\mathbb{E}(L)}. \quad (5.49)$$

$\square$

Proof. For process $\mathbf{Y}$ we have

$$\lim_{m \rightarrow \infty} \frac{1}{m} H(Y_1^m) = \lim_{m \rightarrow \infty} \frac{1}{m} H(X_1^m) = H(\mathbf{X}). \quad (5.50)$$

From the definition of the stopping rule, we have

$$P(Y_1^{M_n}) \leq Q(U_1^n) \leq P(Y_1^{M_n-1}). \quad (5.51)$$

Using the same technique as in the proof of Lemma 5.2.3, we have

$$-\frac{1}{n} \log_2 P(Y_1^{M_n}) = \frac{M_n - \log_2 P(Y_1^{M_n})}{M_n} \rightarrow \frac{H(\mathbf{X})}{\mathbb{E}(L)} \quad \text{a.s.} \quad (5.52)$$

$$-\frac{1}{n} \log_2 P(Y_1^{M_n-1}) = \frac{M_n - 1 - \log_2 P(Y_1^{M_n-1})}{M_n - 1} \rightarrow \frac{H(\mathbf{X})}{\mathbb{E}(L)} \quad \text{a.s..} \quad (5.53)$$

These equations imply that

$$\lim_{n \rightarrow \infty} -\frac{1}{n} \log_2 Q(U_1^n) = \frac{H(\mathbf{X})}{\mathbb{E}(L)} \quad \text{a.s..} \quad (5.54)$$

Since we assume $L(y_{\{i,j\}}) \geq l > 0$, we have

$$\begin{aligned} -\frac{1}{n} \log_2 Q(U_1^n) &\leq -\frac{1}{n} \log_2 P(Y_1^{M_n}) \\ &\leq -\frac{1}{n} \log_2 P_u^n \\ &= -\log_2 P_u. \end{aligned} \quad (5.55)$$

Therefore, by the dominated convergence theorem, we have

$$H(\mathbf{U}) = \lim_{n \rightarrow \infty} -\frac{1}{n} \mathbb{E}(Q(U_1^n)) = \frac{H(\mathbf{X})}{\mathbb{E}(L)}. \quad (5.56)$$

□

The *expansion factor* is defined as the ratio of the expected codeword length to the length of the input source word, namely

$$f = \mathbb{E}(L)/q. \quad (5.57)$$

Remark 5.2.1. In [56, Proposition 2], the author proved that a finite-state word-valued source can be regarded as a word-valued source by providing a method that transforms the finite-state mapping into an one-to-one mapping. Here we give a brief introduction to this method. Given a finite-state mapping ϕ and an i.i.d. source $\mathcal{X}$, we define another mapping $\phi' : \mathcal{X}^* \rightarrow \mathcal{U}^*$. The codebook $\mathcal{Y}'$ of this mapping satisfies the following properties.

- The codeword $y' \in \mathcal{Y}'$ is the concatenation of codewords from ϕ , i.e. $y' = y_1 y_2 \dots y_n, y_i \in \mathcal{Y}$.
- The starting and ending states of y_i satisfy $start(y_{i+1}) = end(y_i) \neq s_0$ when $1 < i < n$, and $start(y_1) = s_0, end(y_n) = s_0$.

We denote by $\mathcal{X}'$ the corresponding input source, i.e., $\mathcal{X}' = \{x^* | \phi(x^*) \in \mathcal{Y}'\}$. Since the graph $\mathcal{F}$ is irreducible, ϕ' has the following properties.

- ϕ' is prefix-free.
- The codebook $\mathcal{Y}'$ is countable.
- $\sum_{y' \in \mathcal{Y}'} P(y') = 1$.

These indicate that the pair $(\mathcal{X}', \phi')$ is a word-valued source with countable codewords. The idea is also used for constructing shaping codes in [5], where the finite-state graph was transformed to its memoryless representation. The normalized geometric Huffman code was applied to the subset of $\mathcal{Y}'$ that consists of codewords with length less than N . Its optimality is guaranteed by letting $N \rightarrow \infty$. $\square$

5.2.4 Asymptotic Normalized KL-Divergence

Similar to the definition of $\hat{P}_u$, the asymptotic probability of occurrence of state $s \in \mathcal{S}$ is defined as

$$\hat{P}_s = \lim_{n \rightarrow \infty} \frac{1}{n} \mathbb{E}(N_s(U_1^n)). \quad (5.58)$$

For every $s_i \in \mathcal{S}$, let $U(s_i) \subset \mathcal{U}$ be the set of edges starting from state s_i . It is easy to prove that

$$\sum_{u \in U(s_i)} N_u(u_1^n) \leq N_{s_i}(u_1^n) \leq \sum_{u \in U(s_i)} N_u(u_1^n) + 1. \quad (5.59)$$

This implies that

$$\hat{P}_{s_i} = \sum_{u \in U(s_i)} \hat{P}_u. \quad (5.60)$$

For the rest of this chapter, we write this equation as $\hat{P}_{s_i} = \sum_j \hat{P}_{u_{ij}}$ for convenience.

Now consider a finite-order Markov process $\hat{\mathbf{U}}$ associated with graph $\mathcal{H}$ and transition probabilities

$$t_{ij}^H = \frac{\hat{P}_{u_{ij}}}{\hat{P}_{s_i}}. \quad (5.61)$$

Denote by $\hat{P}(u_1^n)$ the probability of a length- n sequence generated by this process. To measure the difference between $\hat{\mathbf{U}}$ and $\mathbf{U}$, we define the asymptotic normalized KL-divergence as follows.

$$\lim_{n \rightarrow \infty} \frac{1}{n} D(U_1^n || \hat{U}_1^n) = \lim_{n \rightarrow \infty} \sum_{u_1^n \in \mathcal{U}^n} Q(u_1^n) \log_2 \frac{Q(u_1^n)}{\hat{P}(y_1^n)}. \quad (5.62)$$

The relationship between processes $\mathbf{U}$ and $\hat{\mathbf{U}}$ is summarized in the following lemma.

Lemma 5.2.5. *The asymptotic normalized KL-divergence between processes $\mathbf{U}$ and $\hat{\mathbf{U}}$ satisfies*

$$\lim_{n \rightarrow \infty} \frac{1}{n} D(U_1^n || \hat{U}_1^n) = H(\hat{\mathbf{U}}) - H(\mathbf{U}). \quad (5.63)$$

Specifically,

$$H(\hat{\mathbf{U}}) \geq H(\mathbf{U}). \quad (5.64)$$

□

Proof. We rewrite KL-divergence as

$$\begin{aligned} \frac{1}{n} D(U_1^n || \hat{U}_1^n) &= \frac{1}{n} \sum_{u_1^n \in \mathcal{U}^n} Q(u_1^n) \log_2 \frac{Q(u_1^n)}{\hat{P}(u_1^n)} \\ &= \frac{1}{n} \left[\sum_{u_1^n \in \mathcal{U}^n} Q(u_1^n) \log_2 Q(u_1^n) - \sum_{u_1^n \in \mathcal{U}^n} Q(u_1^n) \log_2 \hat{P}(u_1^n) \right]. \end{aligned} \quad (5.65)$$

The first term on the right side of this equation is the definition of the entropy rate of process $\mathbf{U}$.

The second term can be rewritten as

$$\begin{aligned} \frac{1}{n} \sum_{u_1^n \in \mathcal{U}^n} Q(u_1^n) \log_2 \hat{P}(u_1^n) &= \frac{1}{n} \sum_{u_1^n \in \mathcal{U}^n} Q(u_1^n) \log_2 \prod_{u \in \mathcal{U}} \left(\frac{\hat{P}_u}{\hat{P}_{s_i}} \right)^{N_u(u_1^n)} \\ &= \frac{1}{n} \sum_{u_1^n \in \mathcal{U}^n} Q(u_1^n) \sum_{u \in \mathcal{U}} N_u(u_1^n) \log_2 \frac{\hat{P}_u}{\hat{P}_{s_i}} \\ &= \sum_{u \in \mathcal{U}} \log_2 \frac{\hat{P}_u}{\hat{P}_{s_i}} \sum_{u_1^n \in \mathcal{U}^n} \frac{1}{n} Q(u_1^n) N_u(u_1^n). \end{aligned} \quad (5.66)$$

Combining Lemma 5.2.3 and equation (5.66), we have

$$\begin{aligned} \lim_{n \rightarrow \infty} \frac{1}{n} D(U_1^n || \hat{U}_1^n) &= - \sum_{u \in \mathcal{U}} \log_2 \frac{\hat{P}_u}{\hat{P}_{s_i}} \lim_{n \rightarrow \infty} \sum_{u_1^n \in \mathcal{U}^n} \frac{1}{n} Q(u_1^n) N_u(u_1^n) - H(\mathbf{U}) \\ &= - \sum_{u \in \mathcal{U}} \hat{P}_u \log_2 \frac{\hat{P}_u}{\hat{P}_{s_i}} - H(\mathbf{U}) \\ &= H(\hat{\mathbf{U}}) - H(\mathbf{U}). \end{aligned} \quad (5.67)$$

Since KL-divergence is non-negative, we have

$$H(\hat{\mathbf{U}}) \geq H(\mathbf{U}). \quad (5.68)$$

This finishes the proof. □

Remark 5.2.2. When $H(\hat{\mathbf{U}}) = H(\mathbf{U})$, $\lim_{l \rightarrow \infty} \frac{1}{l} D(U_1^l || \hat{U}_1^l) = 0$. Therefore, the codeword process $\mathbf{U}$ approximates the stationary Markov process $\hat{\mathbf{U}}$. □

5.3 Optimal Shaping Codes for Finite-State Costly Channel

In Section 5.1.1, we introduced the type-I capacity of the noiseless finite-state costly channel. In this section, we first analyze shaping codes that minimize the average cost with a given expansion factor. We refer to this minimization problem as the *type-I shaping problem*, and we call shaping codes that achieve the minimum average cost for a given expansion factor *optimal type-I shaping codes*. To calculate the minimum possible average cost, we solve the following optimization problem.

$$\begin{aligned}
& \underset{\hat{P}_{u_{ij}}}{\text{minimize}} && \sum_{ij} \hat{P}_{u_{ij}} w(u_{ij}) \\
& \text{subject to} && H(\hat{\mathbf{U}}) = - \sum_{ij} \hat{P}_{u_{ij}} \log_2 \frac{\hat{P}_{u_{ij}}}{\sum_j \hat{P}_{u_{ij}}} \geq \frac{H(\mathbf{X})}{f} \\
& && \sum_j \hat{P}_{u_{ji}} = \sum_j \hat{P}_{u_{ij}} \\
& && \sum_{ij} \hat{P}_{u_{ij}} = 1.
\end{aligned} \tag{5.69}$$

The Lagrange Multiplier of this optimization problem is ($\beta \geq 0$)

$$\begin{aligned}
\mathcal{LM} = & \sum_{ij} \hat{P}_{u_{ij}} w(u_{ij}) - \beta \left(- \sum_{ij} \hat{P}_{u_{ij}} \log_2 \frac{\hat{P}_{u_{ij}}}{\sum_j \hat{P}_{u_{ij}}} - \frac{H(\mathbf{X})}{f} \right) \\
& + \sum_j \mu_j \left(\sum_j \hat{P}_{u_{ji}} - \sum_j \hat{P}_{u_{ij}} \right) + \alpha \left(\sum_{ij} \hat{P}_{u_{ij}} - 1 \right),
\end{aligned} \tag{5.70}$$

and

$$\frac{\partial \mathcal{LM}}{\partial \hat{P}_{u_{ij}}} = w(u_{ij}) + \beta \log_2 \frac{\hat{P}_{u_{ij}}}{\sum_j \hat{P}_{u_{ij}}} + \mu_i - \mu_j + \alpha. \tag{5.71}$$

The Karush-Kuhn-Tucker (KKT) conditions [11] require that

$$\frac{\partial \mathcal{LM}}{\partial \hat{P}_{u_{ij}}} = 0 \quad \text{for all } i, j, \tag{5.72}$$

$$\beta(-\sum_{ij} \hat{P}_{u_{ij}} \log_2 \frac{\hat{P}_{u_{ij}}}{\sum_j \hat{P}_{u_{ij}}} - \frac{H(\mathbf{X})}{f}) = 0, \quad (5.73)$$

$$\sum_j \hat{P}_{u_{ji}} = \sum_j \hat{P}_{u_{ij}} \quad \text{for all } j, \quad (5.74)$$

$$\sum_{ij} \hat{P}_{u_{ij}} = 1. \quad (5.75)$$

One special case is when $w(u_{ij}) = -\mu_i + \mu_j - \alpha$ for all i, j . Under this condition, we have $\beta = 0$ and the average cost of this code is

$$\begin{aligned} A(\phi) &= \sum_{ij} \hat{P}_{u_{ij}} (-\mu_i + \mu_j - \alpha) \\ &= \sum_{ij} \hat{P}_{u_{ij}} \mu_j - \sum_{ij} \hat{P}_{u_{ij}} \mu_i - \alpha \\ &= \sum_j \mu_j \sum_i \hat{P}_{u_{ij}} - \sum_i \mu_i \sum_j \hat{P}_{u_{ij}} - \alpha \\ &= \sum_j \mu_j \hat{P}_{s_j} - \sum_i \mu_i \hat{P}_{s_i} - \alpha = -\alpha. \end{aligned} \quad (5.76)$$

The optimization problem (5.69) then becomes trivial. Later we will show this condition plays an important role in the analysis of the type-II capacity. Here we assume that the set $\{w(u_{ij})\}$ can't be represented by $-\mu_i + \mu_j - \alpha$. Then we have $\beta \neq 0$ and the KKT condition (5.73) becomes

$$-\sum_{ij} \hat{P}_{u_{ij}} \log_2 \frac{\hat{P}_{u_{ij}}}{\sum_j \hat{P}_{u_{ij}}} - \frac{H(\mathbf{X})}{f} = 0 \Rightarrow H(\hat{\mathbf{U}}) = \frac{H(\mathbf{X})}{f} = H(\mathbf{U}). \quad (5.77)$$

Given the one-step cost-enumerator matrix $D(B)$ introduced in equation (5.5), we denote by $\lambda(B)$ its Perron root and by $E_L = [\hat{P}_{s_i}/v_i]$, $E_R = [v_i]^\top$ the corresponding eigenvectors. E_L and E_R are normalized such that $E_L E_R = 1$. Using the same technique that is used in the proof of [32, Theorem 1], we know that the optimal distribution $\{\hat{P}_{u_{ij}}\}$ that minimizes the average cost is

$$\hat{P}_{u_{ij}} = \frac{\hat{P}_{s_i}}{v_i} \frac{2^{-Bw(u_{ij})}}{\lambda(B)} v_j, \quad (5.78)$$

where B is the solution of the equation

$$H(\hat{\mathbf{U}}) = -\sum_{ij} \hat{P}_{u_{ij}} \log_2 \frac{\hat{P}_{u_{ij}}}{\sum_j \hat{P}_{u_{ij}}} = H(\mathbf{U}) = \frac{H(\mathbf{X})}{f}. \quad (5.79)$$

The minimum average cost is

$$\begin{aligned} A_{\min}(f) &= \sum_{ij} \hat{P}_{u_{ij}} w(u_{ij}) = \sum_{ij} \hat{P}_{u_{ij}} \frac{\log_2 v_j - \log_2 v_i - \log_2 \frac{\hat{P}_{u_{ij}}}{\hat{P}_{s_i}} - \log_2 \lambda(B)}{B} \\ &= -\frac{\sum_{ij} \hat{P}_{u_{ij}} \log_2 \frac{\hat{P}_{u_{ij}}}{\hat{P}_{s_i}}}{B} - \frac{\log_2 \lambda(B)}{B} = \frac{H(\hat{\mathbf{U}})}{B} - \frac{\log_2 \lambda(B)}{B} = \frac{H(\mathbf{X})}{Bf} - \frac{\log_2 \lambda(B)}{B}. \end{aligned} \quad (5.80)$$

If $u_{ij} = -\mu_i + \mu_j - \alpha$, any setting of $\hat{P}_{u_{ij}}$ that satisfies the constraints in (5.69) also satisfies the KKT conditions. To summarize, we have the following theorem regarding the achievable minimum average cost of a shaping codes on noiseless finite-state costly channel.

Theorem 5.3.1. *When edge costs $\{w(u_{ij})\}$ cannot be represented by $w(u_{ij}) = -\mu_i + \mu_j - \alpha$, the average cost of a type-I shaping code $\phi : \mathcal{S} \times \mathcal{X}^q \rightarrow \mathcal{U}^*$ with expansion factor f is lower bounded by*

$$A_{\min}(f) = \sum_{ij} \hat{P}_{u_{ij}} w(u_{ij}) = \frac{H(\mathbf{X})}{Bf} - \frac{\log_2 \lambda(B)}{B}, \quad (5.81)$$

where

$$\hat{P}_{u_{ij}} = \frac{\hat{P}_{s_i}}{v_i} \frac{2^{-Bw(u_{ij})}}{\lambda(B)} v_j, \quad (5.82)$$

$\lambda(B)$ is the Perron root of the matrix $D(B) = \{2^{-Bw(u_{ij})}\}$, $E_L = [\hat{P}_{s_i}/v_i]$ $E_R = [v_i]^\top$ are the corresponding eigenvectors such that $E_L E_R = 1$, and B is the constant such that

$$H(\hat{\mathbf{U}}) = -\sum_{ij} \hat{P}_{u_{ij}} \log_2 \frac{\hat{P}_{u_{ij}}}{\hat{P}_{s_i}} = H(\mathbf{U}) = \frac{H(\mathbf{X})}{f}. \quad (5.83)$$

If $w(u_{ij}) = -\mu_i + \mu_j - \alpha$ for some $\{w(u_{ij})\}$ and α , the average cost for any shaping code is a constant $-\alpha$. $\square$

Remark 5.3.1. When the minimum average cost is achieved, we have $H(\hat{\mathbf{U}}) = H(\mathbf{U})$. As shown in Remark 5.2.2, the codeword sequence approximates a finite-order stationary Markov process with transition probabilities $\{\hat{P}_{u_{ij}}/\hat{P}_{s_i}\}$, in the sense that

$$\lim_{n \rightarrow \infty} \frac{1}{n} D(U_1^n || \hat{U}_1^n) = 0. \quad (5.84)$$

□

Using Theorem 5.3.1, we study shaping codes that minimize average cost per source symbol, which are closely related to the type-II channel capacity introduced in Section 5.1.2. The *total cost* of a shaping code is

$$\begin{aligned} T(\phi) &= \lim_{n \rightarrow \infty} \frac{\mathbb{E}(C(\phi(X^{nq})))}{nq} \\ &= \lim_{n \rightarrow \infty} \frac{\sum_{u_{ij}} \mathbb{E}(N_{u_{ij}}(Y_1^n)) w(u_{ij})}{nq} \\ &= \frac{\sum_{u_{ij}} \mathbb{E}(N_{u_{ij}}) w(u_{ij})}{q} = \frac{\sum_{ij} \hat{P}_{u_{ij}} \mathbb{E}(L) w(u_{ij})}{q} \\ &= f \sum_{ij} \hat{P}_{u_{ij}} w(u_{ij}). \end{aligned} \quad (5.85)$$

We refer to the problem of minimizing the total cost as the *type-II shaping problem*. Shaping codes that achieve the minimum total cost are referred to as *optimal type-II shaping codes*. The corresponding optimization problem is as follows.

$$\begin{aligned} &\underset{\hat{P}_{u_{ij}}, f}{\text{minimize}} && f \sum_{ij} \hat{P}_{u_{ij}} w(u_{ij}) \\ &\text{subject to} && H(\hat{\mathbf{U}}) \geq H(\mathbf{U}) = \frac{H(\mathbf{X})}{f} \\ &&& \sum_j \hat{P}_{u_{ij}} = \sum_j \hat{P}_{u_{ji}} \\ &&& \sum_{ij} \hat{P}_{u_{ij}} = 1. \end{aligned} \quad (5.86)$$

We divide this optimization problem into two steps. First we fix the expansion factor. The minimum achievable total cost of a shaping code with fixed expansion factor is $T_{\min}(f) = fA_{\min}(f)$. This is already solved in Theorem 5.3.1. Here we study $T_{\min}(f)$ and determine the optimal expansion factor.

It was shown in [35] that the Perron root $\lambda(B)$ is continuous, monotone decreasing and log-convex. However, a sufficient condition for $\lambda(B)$ to be strictly log-convex was not carefully studied. For example, the Perron root of the following matrix

$$D(B) = \begin{bmatrix} 2^{-2B} & 2^{-B} \\ 2^{-3B} & 2^{-2B} \end{bmatrix} \quad (5.87)$$

is $\lambda(B) = 2^{-2B+1}$, which is not strictly log-convex. The following lemma shows a sufficient condition for $\lambda(B)$ to be strictly log-convex.

Lemma 5.3.2. *If edge costs $\{w(u_{ij})\}$ cannot be represented by $w(u_{ij}) = -\mu_i + \mu_j - \alpha$, $\lambda(B)$ is a strictly log-convex function.*

Proof. To prove Lemma 5.3.2, we need the following two lemmas.

Lemma 5.3.3. *The following two statements are equivalent:*

- 1) $w(u_{ij}) = -\mu_i + \mu_j - \alpha$ for some $\{\mu_i\}$ and α .
- 2) *If two different paths from state s_i to state s_j have the same length, their costs are also the same.*

Proof. ($\Rightarrow$) The proof that 1) implies 2) is trivial. For any path $s_i \rightarrow s_{k_1} \rightarrow s_{k_2} \rightarrow \dots \rightarrow s_{k_n} \rightarrow s_j$,

its cost

$$\begin{aligned}
C &= u_{ik_1} + u_{k_1k_2} + \dots + u_{k_{n-1}k_n} + u_{k_nj} \\
&= -\mu_i + \mu_{k_1} - \alpha - \mu_{k_1} + \mu_{k_2} - \alpha - \dots - \mu_{k_{n-1}} + \mu_{k_n} - \alpha - \mu_{k_n} + \mu_j - \alpha \\
&= -\mu_i + \mu_j - (n+2)\alpha
\end{aligned} \tag{5.88}$$

is a function of path length. ($\Leftarrow$) First consider a cycle $\mathcal{L}_1$ that starts and ends at state s_1 . Assume that this cycle has length m and has cost C_1 , and define

$$\alpha = -\frac{C_1}{m}. \tag{5.89}$$

For any cycle $\mathcal{L}_i$ that starts and ends at state s_i with length m' , denote by C_i its cost. Consider the following two paths that start from s_1 and end at s_i .

- Go through $\mathcal{L}_1$ m' times, then go from s_1 to s_i .
- Go from s_1 to s_i using the same path, then go through $\mathcal{L}_i$ m times.

These two paths have the same number of steps, thus they have the same cost. Therefore we have

$$m'C_0 = mC_i \Rightarrow C_i = -m'\alpha. \tag{5.90}$$

This implies that any length- m cycle in the graph has cost $-m\alpha$.

Assume that $\mu_1 = 0$. Now consider paths that start from state s_1 and visit its neighboring states $s_{a_1}, s_{a_2}, \dots$. We can define

$$\mu_{a_i} = w(u_{1a_i}) + \mu_1 + \alpha. \tag{5.91}$$

The paths then visit states $s_{b_1}, s_{b_2}, \dots$, which are neighbors of s_{a_i} and have not been visited before.

Similarly, we can define

$$\mu_{b_j} = w(u_{a_i b_j}) + \mu_{a_i} + \alpha. \quad (5.92)$$

Since this graph is irreducible, we will eventually visit all states. Denote by $\mathcal{L}_1^i$ the path that starts from state 1 and ends at state i and assume the length of this path is n_i . By definition, the cost of this path is $-\mu_1 + \mu_i - n_i\alpha$. For any edge u_{ij} , denote by $\mathcal{L}_j^1$ an arbitrary path from s_j to s_1 with length n'_j . $\mathcal{L}_1^j$ and $\mathcal{L}_j^1$ form a cycle with length $n_j + n'_j$. Therefore the cost of $\mathcal{L}_j^1$ is

$$-(n_j + n'_j)\alpha - (-\mu_1 + \mu_j - n_j\alpha) = -\mu_j + \mu_1 - n'_j\alpha. \quad (5.93)$$

$\mathcal{L}_1^i$, $\mathcal{L}_j^1$ and u_{ij} form a cycle with length $n_i + n'_j + 1$. Therefore its total cost is $-(n_i + n'_j + 1)\alpha$ and

$$\begin{aligned} w(u_{ij}) &= -(n_i + n'_j + 1)\alpha - (-\mu_1 + \mu_i - n_i\alpha) - (-\mu_j + \mu_1 - n'_j\alpha) \\ &= -\mu_i + \mu_j - \alpha. \end{aligned} \quad (5.94)$$

This finishes the proof. □

The next lemma is stated in [71, Theorem 1.37].

Lemma 5.3.4. *If an irreducible matrix $D(B) = \{d_{ij}(B)\}$ satisfies the following two conditions:*

- *every entry $d_{ij}(B)$ is either a constant 0 or a non-negative log-convex function,*
- *at least one entry is a strictly log-convex function,*

then its Perron root $\lambda(B)$ is strictly log-convex. □

Since $\{w(u_{ij})\}$ can't be represented by $-\mu_i + \mu_j - \alpha$, by Lemma 5.3.3, there exists at least two length- n paths from s_i to s_j that have distinct costs for some n . The (i, j) th entry of the matrix $D(B)^n$ has the form

$$d_{ij}^n = 2^{-BC_1} + 2^{-BC_2} + \dots + 2^{-BC_m}, \quad (5.95)$$

where 2^{-BC_k} represents a length- n path from s_i to s_j that has cost C_k . The second-order derivative of $\log_2(d_{ij}^n)$ is

$$\frac{d^2 \log_2(d_{ij}^n)}{dB^2} = \frac{\ln 2 \sum_{x < y} (C_x - C_y)^2 2^{-B(C_x + C_y)}}{(\sum 2^{-BC_z})^2}. \quad (5.96)$$

Since $C_x \neq C_y$ for some x, y , we have $\frac{d^2 \log_2(d_{ij}^n)}{dB^2} > 0$ and therefore d_{ij}^n is strictly log-convex. Then by Lemma 5.3.4, the Perron root of $D(B)^n$, which is $\lambda(B)^n$, is strictly log convex. This implies that $\lambda(B)$ is strictly log-convex. $\square$

We first assume that $\{w(u_{ij})\}$ can't be represented by $-\mu_i + \mu_j - \alpha$. Equation (5.80) can be written as

$$\begin{aligned} A_{\min}(f) &= \sum_{ij} \hat{P}_{u_{ij}} w(u_{ij}) \\ &= \sum_{ij} \frac{\hat{P}_{s_i}}{v_i} \frac{w(u_{ij}) 2^{-Bw(u_{ij})}}{\lambda(B)} v_j \\ &= -\frac{E_L \frac{dD(B)}{dB} E_R}{\lambda(B) \ln 2}. \end{aligned} \quad (5.97)$$

Since E_L is the left eigenvector, we have

$$\begin{aligned} E_L D(B) &= \lambda(B) E_L, \\ \frac{dE_L}{dB} D(B) + E_L \frac{dD(B)}{dB} &= \frac{d\lambda(B)}{dB} E_L + \lambda(B) \frac{dE_L}{dB}. \end{aligned} \quad (5.98)$$

Multiply E_R to both sides of the equation, we have

$$\begin{aligned} \frac{dE_L}{dB} D(B) E_R + E_L \frac{dD(B)}{dB} E_R &= \frac{d\lambda(B)}{dB} E_L E_R + \lambda(B) \frac{dE_L}{dB} E_R, \\ \frac{dE_L}{dB} \lambda(B) E_R + E_L \frac{dD(B)}{dB} E_R &= \frac{d\lambda(B)}{dB} + \lambda(B) \frac{dE_L}{dB} E_R, \\ E_L \frac{dD(B)}{dB} E_R &= \frac{d\lambda(B)}{dB}. \end{aligned} \quad (5.99)$$

Combine equations (5.97) and (5.99), we have

$$A_{\min}(f) = -\frac{\frac{d\lambda(B)}{dB}}{\lambda(B)\ln 2} = -\frac{d\log_2 \lambda(B)}{dB}. \quad (5.100)$$

Therefore the first-order derivative of $H(\hat{\mathbf{U}})$ is

$$\begin{aligned} \frac{dH(\hat{\mathbf{U}})}{dB} &= \frac{d(BA_{\min}(f) + \log_2 \lambda(B))}{dB} \\ &= A_{\min}(f) + B \frac{dA_{\min}(f)}{dB} + \frac{d\log_2 \lambda(B)}{dB} \\ &= A_{\min}(f) - B \frac{d^2 \log_2 \lambda(B)}{dB^2} - A_{\min}(f) \\ &= -B \frac{d^2 \log_2 \lambda(B)}{dB^2} < 0, \end{aligned} \quad (5.101)$$

and

$$\begin{aligned} \frac{df}{dB} &= \frac{d\frac{H(\mathbf{X})}{H(\hat{\mathbf{U}})}}{dB} = -\frac{H(\mathbf{X})}{H(\hat{\mathbf{U}})^2} \frac{dH(\hat{\mathbf{U}})}{dB} \\ &= \frac{BH(\mathbf{X})}{H(\hat{\mathbf{U}})^2} \frac{d^2 \log_2 \lambda(B)}{dB^2}. \end{aligned} \quad (5.102)$$

Combining equations (5.100), (5.101) and (5.102), we have

$$\begin{aligned} \frac{dT_{\min}(f)}{dB} &= \frac{df}{dB} A_{\min}(f) + f \frac{dA_{\min}(f)}{dB} \\ &= \frac{H(\mathbf{X})}{H(\hat{\mathbf{U}})^2} \frac{dH(\hat{\mathbf{U}})}{dB} \frac{d\log_2 \lambda(B)}{dB} - \frac{H(\mathbf{X})}{H(\hat{\mathbf{U}})} \frac{d^2 \log_2 \lambda(B)}{dB^2} \\ &= -\frac{H(\mathbf{X})}{H(\hat{\mathbf{U}})^2} \frac{d^2 \log_2 \lambda(B)}{dB^2} \log_2 \lambda(B). \end{aligned} \quad (5.103)$$

Since $\lambda(B)$ is strictly log-convex, by combining equations (5.102) and (5.103) we have

$$\frac{dT_{\min}(f)}{df} = \frac{-\log_2 \lambda(B)}{B}. \quad (5.104)$$

Starting with $B = 0$, since we assume graph $\mathcal{H}$ has positive capacity and $D(0)$ is the adjacency

matrix of $\mathcal{H}$, we have $\lambda(0) > 1$ and $\frac{dT_{\min}(f)}{df}|_{B \rightarrow 0^+} < 0$. The minimum of $T_{\min}(f)$ will be achieved when $\lambda(B) = 1$. Here we prove the following lemma, which shows the condition under which $\lambda(B) = 1$ is achievable.

Lemma 5.3.5. $\lambda(B) = 1$ is achievable if and only if the graph $\mathcal{H}$ does not contain a cost-0 cycle. If such a cycle exists, $\lambda(B) > 1$ when $B \geq 0$.

Proof. If there is a cost-0 cycle, denote by $\tilde{\mathcal{S}} = \{s_{i_1}, s_{i_2}, \dots, s_{i_n}\}$ the set of states that are on the cycle. Each edge on this cycle has cost $w(u_{i_k i_{k+1}}) = 0$ and therefore $d_{i_k i_{k+1}} = 2^{-Bu_{i_k i_{k+1}}} = 1$. Consider the rows $i_1, i_2, \dots, i_n$ of $D(B)$. From $D(B)E_R = \lambda(B)E_R$ we have

$$\begin{aligned} \sum_j d_{i_k j} v_j &= \lambda(B) v_{i_k} \\ \Rightarrow v_{i_{k+1}} + \sum_{j \neq i_{k+1}} d_{i_k, j} v_j &= \lambda(B) v_{i_k} \\ \Rightarrow \begin{cases} v_{i_{k+1}} + I_{i_k} = \lambda(B) v_{i_k} \\ v_{i_1} + I_{i_n} = \lambda(B) v_{i_n} \end{cases} \end{aligned} \quad (5.105)$$

Assume that except for $u_{i_k i_{k+1}}$ there is an edge that connects s_{i_k} with another state that is not included in $\tilde{\mathcal{S}}$. This edge always exists, otherwise, the graph is not irreducible or has capacity 0. Then we have $I_{i_k} > 0$. By adding up equations (5.105) for all $s_{i_k} \in \tilde{\mathcal{S}}$, we have

$$\sum_k v_{i_k} + \sum_k I_{i_k} = \lambda(B) \sum_k v_{i_k}. \quad (5.106)$$

Since $\sum_k I_{i_k} > 0$, $\lambda(B) > 1$ for any $B \geq 0$.

If no cost-0 cycle exists, we consider the matrix $D(B)^{|\mathcal{S}|+1}$, which represents all length- $|\mathcal{S}| + 1$ paths in $\mathcal{H}$. Each path contains a cycle and therefore has a positive cost. This implies that for B sufficiently large, every row sum of $D(B)^{|\mathcal{S}|+1}$ is strictly less than 1 and

$$\lambda(B^{|\mathcal{V}_H|+1}) < 1 \Rightarrow \lambda(B) < 1. \quad (5.107)$$

Since $\lambda(B)$ is continuous, monotone decreasing and $\lambda(0) > 1$, there exists a $B > 0$ such that $\lambda(B) = 1$. $\square$

Remark 5.3.2. From equation (5.104), we know that if there is a cost-0 cycle, $T_{\min}(f)$ is a monotone decreasing function. Therefore no global minimum can be reached. $\square$

Remark 5.3.3. [13] and [62] assumed that the cost of any sequence with length $l \geq |\mathcal{S}|$ is strictly positive. Since there is a cycle inside any sequence with length $\geq |\mathcal{S}|$, it is easy to check the above requirement is equivalent to the no cost-0 cycle condition in Lemma 5.3.5. $\square$

Suppose $u_{ij} = -\mu_i + \mu_j - \alpha$ for some $\{\mu_i\}$ and α . Since the average cost of the shaping code is a constant, minimizing total cost is equivalent to minimizing the expansion factor. Since

$$f = \frac{H(\mathbf{X})}{\mathbf{U}} \geq \frac{H(\mathbf{X})}{\hat{\mathbf{U}}}, \quad (5.108)$$

this is equivalent to maximizing $H(\hat{\mathbf{U}})$ without average cost constraint. This is solved in [32, Theorem 1]. The results rely on matrix $D(0)$ and the minimum total cost is

$$T_{\min} = -\frac{H(\mathbf{X})}{\log_2 \lambda(0)} \alpha. \quad (5.109)$$

Assume that $\lambda(B_0) = 1$ for a given B_0 . We will show that $D(0)$ and $D(B_0)$ deliver the same result, so we don't need to consider this special case separately. Entries of matrix $D(B_0)$ have the following forms:

$$d_{ij}(B_0) = 2^{-B_0 w(u_{ij})} = \frac{u_i \kappa}{u_j}, \quad (5.110)$$

where $u_i = 2^{B_0 \mu_i}$ and $\kappa = 2^{B_0 \alpha}$. The left and right eigenvectors of $D(B_0)$ are

$$E_L(B_0) = [\hat{P}_{s_i}^* / v_i] \quad E_R(B_0) = [v_i]^\top. \quad (5.111)$$

Construct vectors

$$E'_L = [\hat{P}_{s_i}^* u_i / v_i] \quad E'_R = [v_i / u_i]^\top \quad (5.112)$$

based on equations (5.110) and (5.111). It is easy to check that

$$E'_L D(0) = \frac{1}{\kappa} E'_L \quad D(0) E'_R = \frac{1}{\kappa} E'_R. \quad (5.113)$$

This implies that E'_L and E'_R are the left and right eigenvectors of $D(0)$ and the Perron root $\lambda(0) = \frac{1}{\kappa}$. Thus the optimal occurrence probability and minimum total cost when $B = 0$ are

$$\begin{aligned} \hat{P}_{u_{ij}}^* &= \frac{\hat{P}_{s_i}^*}{\frac{v_i}{u_i}} \frac{2^0}{\lambda(0)} \frac{v_j}{u_j} = \frac{\hat{P}_{s_i}^*}{v_i} \frac{u_i \kappa}{u_j} v_j \\ &= \frac{\hat{P}_{s_i}^*}{v_i} 2^{-B_0 w(u_{ij})} v_j, \end{aligned} \quad (5.114)$$

and

$$T(\phi) = -\frac{H(\mathbf{X})}{\log_2 \lambda(0)} \alpha = \frac{H(\mathbf{X})}{B_0}, \quad (5.115)$$

respectively. To summarize, we have the following theorem that determines the minimum achievable total cost of a shaping code.

Theorem 5.3.6. *If the cost-0 cycle does not exist, the minimum total cost of a type-II shaping code $\phi : \mathcal{S} \times \mathcal{X}^q \rightarrow \mathcal{U}^*$ is given by*

$$T_{min} = f^* \sum_{ij} \hat{P}_{u_{ij}}^* w(u_{ij}) = \frac{H(\mathbf{X})}{B^*}, \quad (5.116)$$

where

$$\hat{P}_{u_{ij}}^* = \frac{\hat{P}_{s_i}^*}{v_i} 2^{-B^* w(u_{ij})} v_j, \quad (5.117)$$

B^* is a constant such that $\lambda(B^*) = 1$, and $E_L = [\hat{P}_{s_i}^* / v_i]$ and $E_R = [v_i]^\top$ are the corresponding

eigenvectors such that $E_L E_R = 1$. The corresponding expansion factor f^* is

$$f^* = \frac{H(\mathbf{X})}{-\sum_{ij} \hat{P}_{u_{ij}}^* \log_2 \frac{\hat{P}_{u_{ij}}^*}{\hat{P}_{s_i}^*}} = \frac{H(\mathbf{X})}{B^* \sum_{ij} \hat{P}_{u_{ij}}^* w(u_{ij})}. \quad (5.118)$$

If there is a cost-0 cycle in $\mathcal{H}$, the total cost is a decreasing function of f . $\square$

Remark 5.3.4. Theorem 5.3.6 is an extension of results in [9], [13, Proposition 2.1.] and [35, Theorem 5.1.]. In particular, we establish the relationship between the expansion factor of a type-II shaping code and its minimum possible total cost. We study the one-step cost-enumerator matrix $D(B)$ and find a sufficient condition for its Perron root to be a strictly log-convex function of B . We also study the special case when there is a cost-0 cycle in the graph, in which no global minimum can be reached.

Example 5.3.1. The one-step cost-enumerator matrix for the SLC flash memory channel introduced in Example 5.1.1 has the form:

$$D(B) = \begin{bmatrix} 2^{-B} & 2^{-2B} & 0 & 0 \\ 0 & 0 & 2^{-4B} & 2^{-4B} \\ 2^{-2B} & 2^{-3B} & 0 & 0 \\ 0 & 0 & 2^{-4B} & 2^{-4B} \end{bmatrix}. \quad (5.119)$$

It is easy to check that $\lambda(0.3856) = 1$, and the corresponding left and right eigenvectors are

$$\begin{aligned} E_L &= [0.7788, 0.5961, 0.3117, 0.3117], \\ E_R &= [0.7243, 0.2899, 0.5544, 0.2899]^\top. \end{aligned} \quad (5.120)$$

The optimal symbol distributions that minimize the total cost are shown in Table 5.2. $\square$

Example 5.3.2. Another application of shaping codes is in the area of DNA synthesis [41]. The time of encoding a nitrogenous base (A, C, G, and N), which can be treated as cost, depends on

Table 5.2: Optimal symbol occurrence probabilities for flash memory channel.

u	000	001	010	011	100	101	110	111
$\hat{P}_u$	0.4318	0.1323	0.1135	0.0593	0.1323	0.0405	0.0593	0.0310

the last encoded nitrogenous base. The edge costs are shown in Table 5.3. Suppose the input is a

Table 5.3: Cost table of DNA synthesis.

		Last Encoded DNA symbol			
		A	C	G	N
Encoding DNA symbol	A	4	1	2	3
	C	3	4	1	2
	G	2	3	4	1
	N	1	2	3	4

uniform binary i.i.d. source. By Theorem 5.4.1, the minimum total cost is achieved when

$$\lambda(B) = 1 \Rightarrow B = 0.9468, \quad (5.121)$$

and

$$T_{\min} = \frac{H(\mathbf{X})}{B} = \frac{1}{0.9468} = 1.0562. \quad (5.122)$$

Notice that the symbol costs in each column of Table 5.3 are the permutation of $\{1, 2, 3, 4\}$. This implies that we can design a code for this channel by using a shaping code for a memoryless costly channel. Consider a size-4 alphabet Ω with independent symbol costs 1, 2, 3, and 4. Given a sequence generated by this alphabet and a starting nitrogenous base, we can define a mapping that translates it into a DNA sequence by treating symbols as the differences. For example, if the starting base is A and the first symbol has weight 3, then the next base is C. Similarly, if the next symbol has weight 2, then it will be translated as base N. In [46], it is proved that Varn codes are asymptotically optimal shaping codes for uniform i.i.d. sources. By combining a Varn code with the above mapping, we construct an optimal shaping code for the DNA synthesis. $\square$

5.4 Optimal Shaping Code Design

The DNA synthesis channel in Example 5.3.2 is a special channel, in the sense that the graph is fully connected and $\{w(u_{ij})|u_{ij} \in U(s_i)\} = \{1, 2, 3, 4\}$ for all starting state s_i . In this section, we consider the general problem of designing optimal type-I and type-II shaping codes. First of all, we prove a lemma stating that any type-II shaping problem can be converted to a type-II problem on a modified costly channel, where the edge costs are the logarithm of some probability distribution. We then prove the theorems that establish the equivalence between type-I and type-II shaping codes. The rest of the section is on designing asymptotically optimal shaping codes for noiseless finite-state costly channels and arbitrary i.i.d. sources. We propose the generalized Varn code for a uniform i.i.d. source and prove its optimality. We then prove a separation theorem that extends generalized Varn codes to arbitrary i.i.d. sources.

5.4.1 Equivalence Theorem

We consider the channel with modified edge costs

$$w'(u_{ij}) = -\log_2 \frac{\hat{P}_{u_{ij}}^*}{\hat{P}_{s_i}^*}, \quad (5.123)$$

where $\hat{P}_{u_{ij}}^*, \hat{P}_{s_i}^*$ are given in Theorem 5.3.6. Its one-step cost-enumerator matrix is $D'(B) = \{d'_{ij}(B)\} = \{2^{-Bw'(u_{ij})}\}$. It is easy to check that $\lambda'(1) = 1$, $E'_R(1) = [1]$ and $E'_L(1) = [\hat{P}_{s_i}^*]$. This implies that the optimal type-II shaping codes on the modified channel are also optimal on the original channel, in the sense that the symbol occurrence probabilities $\{\hat{P}_{u_{ij}}\}$ are identical and the minimum total cost is achieved. We can prove the following lemma, which show the equivalence between type-II shaping codes for channels with different edge costs.

Lemma 5.4.1. *Given a noiseless finite-state costly channel with edge costs $\{w(u_{ij})\}$, consider a*

modified channel with edge costs

$$w'(u_{ij}) = -\log_2 \frac{\hat{P}_{u_{ij}}^*}{\hat{P}_{s_i}^*} = B^* w(u_{ij}) + \log_2 v_i - \log_2 v_j, \quad (5.124)$$

where $\hat{P}_{u_{ij}}^*, \hat{P}_{s_i}^*, v_i$ and B^* are calculated in Theorem 5.3.6. If there exists a shaping code $\phi : \mathcal{S} \times \mathcal{X}^q \rightarrow \mathcal{U}^*$ such that

$$f \sum_{ij} \hat{P}_{u_{ij}} w'(u_{ij}) - H(\mathbf{X}) < \delta \quad (5.125)$$

for some $\delta > 0$, the total cost of this code on the original channel satisfies

$$f \sum_{ij} \hat{P}_{u_{ij}} w(u_{ij}) - \frac{H(\mathbf{X})}{B^*} < \frac{\delta}{B^*}. \quad (5.126)$$

Proof. For the modified channel, we have

$$\begin{aligned} f \sum_{ij} \hat{P}_{u_{ij}} w'(u_{ij}) &= f \sum_{ij} \hat{P}_{u_{ij}} (B^* w(u_{ij}) + \log_2 v_i - \log_2 v_j) \\ &= B^* f \sum_{ij} \hat{P}_{u_{ij}} w(u_{ij}) + f \sum_{ij} \hat{P}_{u_{ij}} (\log_2 v_i - \log_2 v_j) \\ &= B^* f \sum_{ij} \hat{P}_{u_{ij}} w(u_{ij}). \end{aligned} \quad (5.127)$$

Therefore the total cost on the original channel satisfies

$$f \sum_{ij} \hat{P}_{u_{ij}} w(u_{ij}) - \frac{H(\mathbf{X})}{B^*} = \frac{f \sum_{ij} \hat{P}_{u_{ij}} w'(u_{ij}) - H(\mathbf{X})}{B^*} < \frac{\delta}{B^*}. \quad (5.128)$$

This finishes the proof. □

Remark 5.4.1. The idea of creating a modified channel with edge costs $\{w'(u_{ij}) = -\log_2 P_{ij}\}$ was also used in [26]. The authors showed that an optimal type-II shaping code generates a codeword sequence that approximates a stationary Markov process with transition probabilities $\{P_{ij}\}$. The criterion for optimality was the vanishing of a form of normalized conditional Kullback-Leibler

(KL) divergence between a subset of codewords of fixed length and words generated with the target Markov process, asymptotically in the block length. $\square$

The next two theorems establish the equivalence between type-I and type-II shaping codes.

Theorem 5.4.2. *Given a noiseless finite-state costly channel with edge costs $\{w(u_{ij})\}$, consider a modified channel with edge costs*

$$w'(u_{ij}) = -\log_2 \frac{\hat{P}_{u_{ij}}}{\hat{P}_{s_i}} = Bw(u_{ij}) + \log_2 v_i - \log_2 v_j + \log_2 \lambda(B), \quad (5.129)$$

where $\hat{P}_{u_{ij}}, \hat{P}_{s_i}, v_i$ and B are calculated in Theorem 5.3.1. For any $\gamma, \eta > 0$, there exists a $\delta > 0$ such that if there exists a shaping code $\phi : \mathcal{S} \times X^q \rightarrow \mathcal{U}^*$ with expansion factor f' such that

$$f' \sum_{ij} \hat{P}'_{u_{ij}} w'(u_{ij}) - H(\mathbf{X}) < \delta, \quad (5.130)$$

the average cost of this code on the original channel is upper bounded by

$$\sum_{ij} \hat{P}'_{u_{ij}} w(u_{ij}) - \left(\frac{H(\mathbf{X})}{Bf} - \frac{\log_2 \lambda(B)}{B} \right) < \gamma \quad (5.131)$$

and the expansion factor of this code f' satisfies $|f' - f| < \eta$.

Proof. The optimal type-II shaping code on the modified channel has expansion factor $f^* = f$ and total cost $T_{\min} = H(\mathbf{X})$. From equation (5.104), we know that $T_{\min}(f') = f' \sum_{ij} \hat{P}'_{u_{ij}} w'(u_{ij})$ is continuous, monotone decreasing on $[H(\mathbf{X})/\log_2 \lambda(0), f^*)$ and monotone increasing on $[f^*, \infty)$. This is shown schematically by the green curve in Fig. 5.3. For any $\gamma, \eta > 0$, let $\eta' = \min\{\eta, \frac{f}{2}, \frac{Bf^2\gamma}{4H(\mathbf{X})}\}$. Then there exists a $\delta' > 0$ such that if there exists a code such that

$$f' \sum_{ij} \hat{P}'_{u_{ij}} w'(u_{ij}) - H(\mathbf{X}) < \delta', \quad (5.132)$$

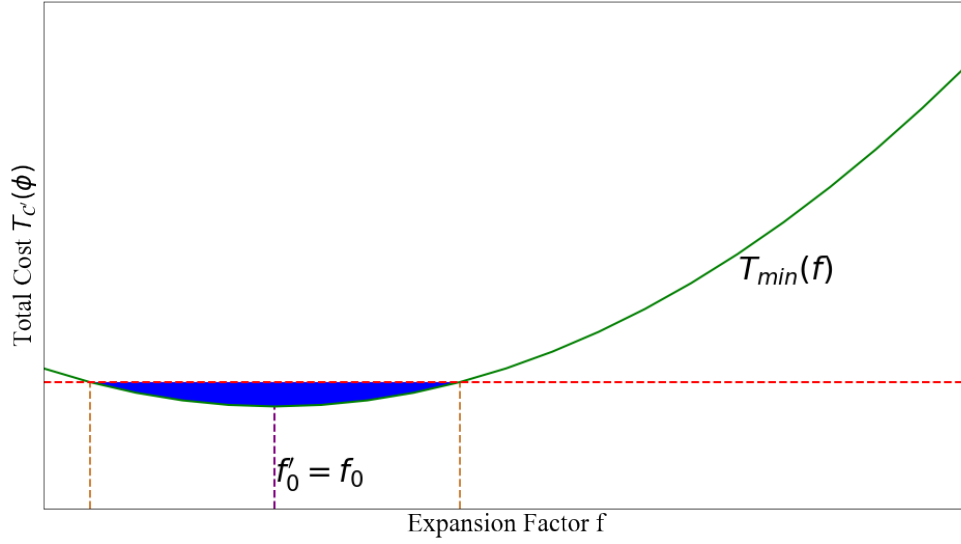


Figure 5.3: Achievable total cost vs. expansion factor.

as indicated by the blue area in Fig. 5.3, the expansion factor of this code $|f' - f| < \eta' \leq \eta$. By combining equations (5.127) and (5.132), the average cost of this code on the original channel satisfies

$$\begin{aligned} B \sum_{ij} \hat{P}'_{u_{ij}} w(u_{ij}) + \log_2 \lambda(B) - \frac{H(\mathbf{X})}{f'} &< \frac{\delta'}{f'} \\ \Rightarrow \sum_{ij} \hat{P}'_{u_{ij}} w(u_{ij}) &< \frac{H(\mathbf{X}) + \delta'}{Bf'} - \frac{\log_2 \lambda(B)}{B}. \end{aligned} \quad (5.133)$$

Let $\delta = \min\{\delta', \frac{Bf\gamma}{4}\}$. If there exists a shaping code with total cost upper bounded by $H(\mathbf{X}) + \delta$, the average cost of this code on the original channel satisfies

$$\begin{aligned} \sum_{ij} \hat{P}'_{u_{ij}} w(u_{ij}) &< \frac{H(\mathbf{X}) + \delta}{Bf'} - \frac{\log_2 \lambda(B)}{B} \\ &< \frac{H(\mathbf{X}) + \delta}{B(f - \eta')} - \frac{\log_2 \lambda(B)}{B} \\ &= \frac{H(\mathbf{X})}{Bf} + \frac{H(\mathbf{X})}{Bf} \frac{\eta'}{f - \eta'} + \frac{\delta}{B(f - \eta')} - \frac{\log_2 \lambda(B)}{B} \\ &< \frac{H(\mathbf{X})}{Bf} + \frac{H(\mathbf{X})}{Bf} \frac{2\eta'}{f} + \frac{2\delta}{Bf} - \frac{\log_2 \lambda(B)}{B} \\ &< \frac{H(\mathbf{X})}{Bf} - \frac{\log_2 \lambda(B)}{B} + \gamma. \end{aligned} \quad (5.134)$$

This finishes the proof. $\square$

Theorem 5.4.3. *Given a noiseless finite-state costly channel that doesn't contain a cost-0 cycle. Denote by B^* the constant such that $\lambda(B^*) = 1$ and by f^* the expansion factor of an optimal type-II shaping code. For any $\gamma > 0$, there exists $\delta, \eta > 0$ such that if a shaping code $\phi : \mathcal{S} \times \mathcal{X}^q \rightarrow \mathcal{U}^*$ with expansion factor f' satisfies*

$$\sum_{ij} \hat{P}'_{u_{ij}} w(u_{ij}) - A_{\min}(f') < \delta, \quad |f' - f^*| < \eta, \quad (5.135)$$

the total cost of this code satisfies

$$f' \sum_{ij} \hat{P}_{u_{ij}} w(u_{ij}) - \frac{H(\mathbf{X})}{B^*} < \gamma. \quad (5.136)$$

Proof. We first study the properties of

$$I_{\min}(f) = A_{\min}(f) - \frac{H(\mathbf{X})}{B^* f}. \quad (5.137)$$

The first derivative of $I_{\min}(f)$ is

$$\begin{aligned} \frac{dI_{\min}(f)}{df} &= \frac{\frac{dT_{\min}(f)}{df}}{f} - \frac{B^* T_{\min}(f) - H(\mathbf{X})}{B^* f^2} \\ &= -\frac{\log_2 \lambda(B)}{Bf} - \frac{H(\mathbf{X})}{Bf^2} + \frac{\log_2 \lambda(B)}{Bf} + \frac{H(\mathbf{X})}{B^* f^2} \\ &= \frac{H(\mathbf{X})}{f^2} \left(\frac{1}{B^*} - \frac{1}{B} \right). \end{aligned} \quad (5.138)$$

Therefore $I_{\min}(f)$ is continuous, monotone decreasing on $[H(\mathbf{X})/\log_2 \lambda(0), f^*)$ (or $B \in [0, B^*)$), continuous, monotone increasing on $[f^*, \infty)$ (or $B \in [B^*, \infty)$) and

$$I_{\min}(f^*)|_{B=B^*} = \frac{H(\mathbf{X})}{B^* f^*} - \frac{\log_2 \lambda(B^*)}{B^*} - \frac{H(\mathbf{X})}{B^* f^*} = 0. \quad (5.139)$$

This implies that for any $0 < \delta' < \frac{\gamma}{3f^*}$, there exists a $\eta' > 0$ such that if $|f' - f^*| < \eta'$, $I_{\min}(f') < \delta'$. Let $\delta = \delta'$, $\eta = \min\{\eta', f^*/2\}$. If there exists a shaping code such that $|f' - f^*| < \eta$, $\sum_{ij} \hat{P}_{uij} w(u_{ij}) - A_{\min}(f') < \delta$, we have

$$\sum_{ij} \hat{P}'_{uij} w(u_{ij}) < \delta + A_{\min}(f') < \delta + \delta' + \frac{H(\mathbf{X})}{B^* f'} < \frac{2\gamma}{3f^*} + \frac{H(\mathbf{X})}{B^* f'}. \quad (5.140)$$

The total cost of this code satisfies

$$\begin{aligned} T(\phi) &= f' \sum_{ij} \hat{P}'_{uij} w(u_{ij}) < (f^* + \eta) \frac{2\gamma}{3f^*} + \frac{H(\mathbf{X})}{B^*} \\ &\leq \frac{H(\mathbf{X})}{B^*} + \gamma. \end{aligned} \quad (5.141)$$

This finishes the proof. $\square$

5.4.2 Generalized Varn Code

In this subsection, we propose an asymptotically optimal type-II shaping code for uniform i.i.d. source. Our scheme was inspired by Varn codes, which were proposed in [78] and were used to construct shaping codes for memoryless costly channels in [48]. Given a uniform i.i.d. input source $\mathcal{X}$, a *generalized Varn Code* on the noiseless finite-state costly channel is a collection of tree-based variable-length mappings $\phi : \mathcal{S} \times \mathcal{X}^q \rightarrow \mathcal{U}^*$. Denote by $\mathcal{J}_k$ the set of codewords starting from state s_k , namely

$$\mathcal{J}_k = \{\phi(s_k, x^q) | x^q \in \mathcal{X}^q\}. \quad (5.142)$$

Codewords in $\mathcal{J}_k$ are generated according to the following steps.

- Set state $s_k \in \mathcal{S}$ as the root of the tree.
- Expand the root node. The edge costs $\{w'(u_{kl})\}$ are defined in Lemma 5.4.1. The cost of a leaf node is the cost of the path from root node to the leaf node.

- Expand the leaf node that has the lowest cost.
- Repeat the previous steps until the total number of leaf nodes $M \geq |\mathcal{X}|^q$. Denoting by U_0 the largest outdegree in $\mathcal{H}$, we have $M \leq |\mathcal{X}|^q + U_0$. The cost of the leaf code is denoted by C_m^k , $1 \leq m \leq M$. Delete the leaf nodes that have the largest cost until the number of leaf nodes equals to $|\mathcal{X}|^q$. Each path from the root node s_k to the leaf node represents one codeword in $\mathcal{Y}_k$.

The following lemma gives an upper bound on the total cost of the generalized Varn codes.

Lemma 5.4.4. *The total cost of the generalized Varn code $\phi : \mathcal{S} \times \mathcal{X}^q \rightarrow \mathcal{U}^*$ is upper bounded by*

$$T(\phi) \leq \frac{\log_2 M}{q} + \frac{\max_{ij} \{w'(u_{ij})\}}{q} \rightarrow \log_2 |\mathcal{X}| \quad \text{as } q \rightarrow \infty. \quad (5.143)$$

Proof. For the tree that generates $\mathcal{Y}_k$, we consider the internal node that was the last one to be expanded and suppose it has cost C_{int}^k . The cost of any leaf node is larger than C_{int}^k , so

$$C_m^k \geq C_{\text{int}}^k. \quad (5.144)$$

Since this internal node is the last one to be expanded, its cost is greater than that of any other internal nodes, implying

$$C_m^k \leq C_{\text{int}}^k + \max_{ij} \{w'(u_{ij})\}. \quad (5.145)$$

By definition, $\sum_j 2^{-w'(u_{ij})} = 1$ for all i , therefore we have

$$\sum_{m=1}^M 2^{-C_m^k} = 1. \quad (5.146)$$

Combining equations (5.144) and (5.146), we have

$$1 = \sum_i 2^{-C_m^k} \leq M 2^{-C_{\text{int}}^k} \Rightarrow C_{\text{int}}^k \leq \log_2 M. \quad (5.147)$$

Equation (5.145) then implies that

$$C_m^k \leq C_{\text{int}}^k + \max_{ij} \{w'(u_{ij})\} \leq \log_2 M + \max_{ij} \{w'(u_{ij})\}. \quad (5.148)$$

Since all codewords are upper bounded, we have

$$\begin{aligned} T(\phi) &\leq \frac{\max\{C_m^k\}}{q} \leq \frac{\log_2 M}{q} + \frac{\max_{ij} \{w'(u_{ij})\}}{q} \\ &\rightarrow \log_2 |\mathcal{X}| \quad \text{as } q \rightarrow \infty. \end{aligned} \quad (5.149)$$

□

Remark 5.4.2. By extending some leaf nodes to states that are not visited by the original code, we can make graph $\mathcal{G}_0$ a complete graph. Then we can choose any state as the starting state. This operation only adds a constant to the cost of a codeword and therefore doesn't affect the asymptotic performance of the generalized Varn codes.

Remark 5.4.3. In [61], the author showed that Varn codes and Tunstall codes [75] are identical under certain conditions. This revealed the dual nature between shaping code for noiseless costly channel and source coding. Tunstall codes have been extended to design variable to fixed-length codes for Markov sources. For more details, see [63] and [74].

Example 5.4.1. For the SLC flash memory channel introduced in Example 5.1.1, the modified edge costs are shown in Table 5.4. We construct generalized Varn codes with different codebook sizes. The total cost for each $\mathcal{Y}_k$ is shown in Fig. 5.4 and the total cost on the SLC flash channel as a function of codebook size is shown in Fig. 5.5.

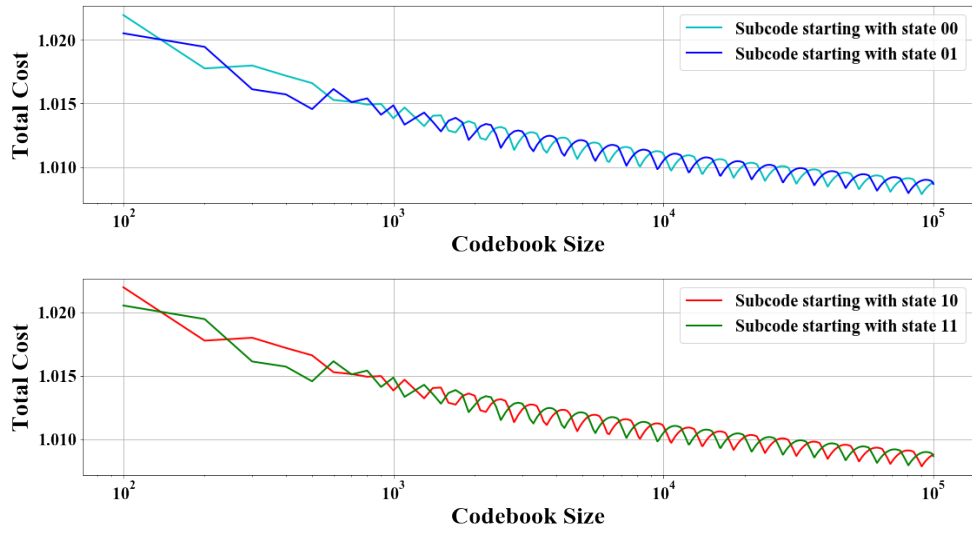


Figure 5.4: The total cost of the subcodebooks of a generalized Varn code on the modified channel with different starting state.

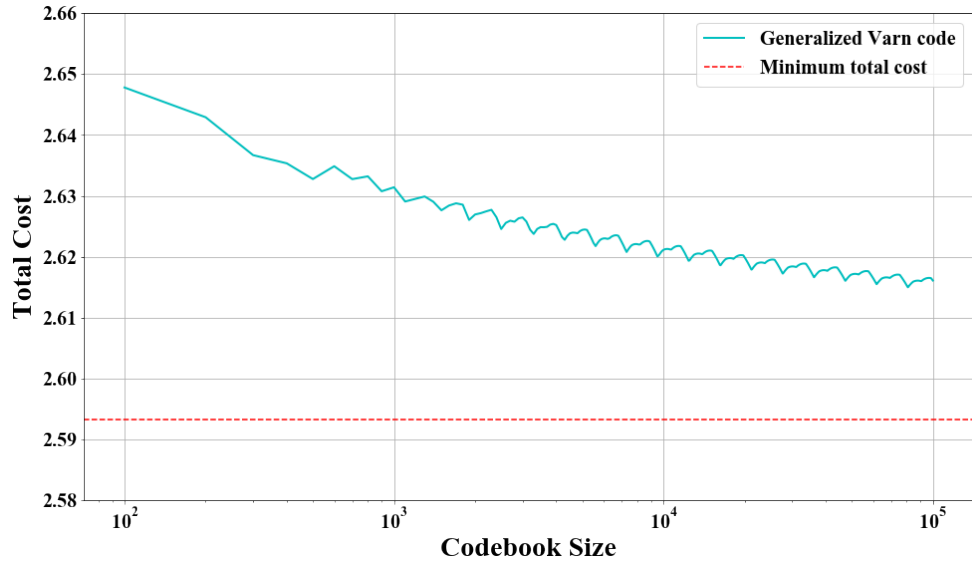


Figure 5.5: The total cost of a generalized Varn code on the original SLC flash memory channel

Table 5.4: Modified cost for flash memory channel.

u	000	001	010	011	100	101	110	111
$C(u)$	0.3805	2.0923	0.6068	1.5423	0.3855	2.0923	0.6068	1.5423

5.4.3 Separation Theorem

We now present a separation theorem for type-II shaping codes. It states that the minimum total cost can be achieved by a concatenation of optimal lossless compression with an optimal type-II shaping code for a uniform i.i.d. source.

Theorem 5.4.5. *Given an i.i.d. source $\mathbf{X}$ and a noiseless finite-state costly channel with edge costs $\{w(u_{ij})\}$, the minimum total cost can be achieved by a concatenation of an optimal lossless compression code with an optimal type-II shaping code for a uniform i.i.d. source.*

Proof. The proof is similar to the proof of Theorem 3.3.4. The only difference is that the inner code of the concatenation, which was a modified Varn code, is replaced by the generalized Varn code introduced in Section 5.4.2. $\square$

By combining Theorems 5.4.5, 5.4.2 and 5.4.3, we can conclude the separation theorem for type-I shaping codes.

Theorem 5.4.6. *Given the i.i.d. source $\mathbf{X}$, the noiseless finite-state costly channel with edge costs $\{w(u_{ij})\}$, and the expansion factor f , the minimum average cost can be achieved by a concatenation of an optimal lossless compression code with a binary optimal type-I shaping code for uniform i.i.d. source and expansion factor*

$$f' = \frac{f}{H(\mathbf{X})}. \quad (5.150)$$

$\square$

5.5 Markov Distribution Matching Code Design

Given the target transition probabilities $\{P_{ij}\}$, Markov distribution matching (MDM) considers the problem of mapping an i.i.d. sequence of source symbols to an output sequence of symbols that approximates a finite-order Markov process with transition probabilities $\{P_{ij}\}$. An optimal MDM code must satisfy two conditions: the codeword sequence has symbol occurrence probabilities $\{\hat{P}_{u_{ij}}\}$ such that $\hat{P}_{u_{ij}}/\hat{P}_{s_i} = P_{ij}$, and the output sequence looks like a finite-order Markov process. We measure the performance of MDM code using the asymptotic normalized KL-divergence defined in Lemma 5.2.5.

It has been shown in Theorems 5.4.2 and 5.3.6 that an optimal shaping code generates an output sequence that approximates a Markov process with transition probabilities $\{\hat{P}_{u_{ij}}/\hat{P}_{s_i}\}$, in the sense that $\lim_{n \rightarrow \infty} \frac{1}{n} D(U_1^n || \hat{U}_1^n) = 0$. This implies that we can solve the distribution matching problem by designing a corresponding shaping code. In this section, we consider the problem of designing optimal MDM codes. We first formulate the problem of *generating a finite-order Markov process* and then show the connection between MDM codes and shaping codes. We then propose a *generalized expansion factor* to measure the performance of a MDM code.

5.5.1 Problem Formulation

We use the asymptotic normalized Kullback-Leibler divergence to formally define an optimal MDM code ϕ for transition probabilities $\{P_{ij}\}$.

Definition 5.5.1. A finite-state variable-length code $\phi : \mathcal{S} \times \mathcal{X}^q \rightarrow \mathcal{U}^*$ is an optimal MDM code for transition probabilities $\{P_{ij}\}$ if

$$\lim_{n \rightarrow \infty} \frac{1}{n} D(U_1^n || \tilde{U}_1^n) = 0, \quad (5.151)$$

where $\tilde{U}$ is a finite-order Markov process with transition probabilities $\{P_{ij}\}$. □

By combining equation (5.56) and Lemma 5.2.5, we can prove the following lemma.

Lemma 5.5.1. *The expansion factor of a shaping code satisfies the lower bound*

$$f = \frac{H(\mathbf{X})}{H(\mathbf{U})} \geq \frac{H(\mathbf{X})}{H(\hat{\mathbf{U}})} \quad (5.152)$$

with equality if and only if $\lim_{n \rightarrow \infty} \frac{1}{n} D(U_1^n || \hat{U}_1^n) = H(\hat{\mathbf{U}}) - H(\mathbf{U}) = 0$. When $f = H(\mathbf{X})/H(\hat{\mathbf{U}})$, this code is an optimal MDM code for transition probabilities $\{\hat{P}_{u_{ij}}/\hat{P}_{s_i}\}$. $\square$

As in the derivation of (5.67) in Lemma 5.2.5, we find

$$\begin{aligned} D(f) &= \lim_{n \rightarrow \infty} \frac{1}{n} D(U_1^n || \tilde{U}_1^n) = - \sum_{ij} \hat{P}_{u_{ij}} \log_2 P_{ij} - H(\mathbf{U}) \\ &= - \sum_{ij} \hat{P}_{u_{ij}} \log_2 P_{ij} - \frac{H(\mathbf{X})}{f} = \frac{-f \sum_{ij} \hat{P}_{u_{ij}} \log_2 P_{ij} - H(\mathbf{X})}{f}. \end{aligned} \quad (5.153)$$

From Theorem 5.3.6, we know that for a channel with edge costs $\{w(u_{ij}) = -\log_2 P_{ij}\}$, the minimum total cost is $T_{\min} = H(\mathbf{X})$. The shaping code that achieves this lower bound has the following two properties.

- The probability of occurrence $\hat{P}_{u_{ij}}$ of symbol u_{ij} satisfies $\hat{P}_{u_{ij}}/\hat{P}_{s_i} = P_{ij}$ for all u_{ij} ,
- The normalized KL distance between $\mathbf{U}$ and $\hat{\mathbf{U}}$ satisfies

$$\lim_{n \rightarrow \infty} \frac{1}{n} D(U_1^n || \hat{U}_1^n) = 0. \quad (5.154)$$

This implies that this code generates a sequence that approximates a finite-order Markov process with transition distribution $\{P_{ij}\}$. This analysis also implies the expansion factor of an optimal MDM code. We have thus established the following relationship between optimal shaping codes and optimal Markov distribution matching codes, extending the result in [26].

Theorem 5.5.2. *The optimal type-II shaping code with edge costs $\{w(u_{ij})\}$, or the equivalent type-I shaping code from Theorem 5.4.2, is an optimal MDM code for transition probabilities $\{P_{ij}\}$ if*

$$w(u_{ij}) = -\log_2 P_{ij} \quad (5.155)$$

for every symbol u_{ij} , in the sense that

$$\lim_{n \rightarrow \infty} \frac{1}{n} D(U_1^n || \tilde{U}_1^n) = 0. \quad (5.156)$$

The expansion factor of this optimal MDM code is

$$f^* = \frac{H(\mathbf{X})}{-\sum_i \hat{P}_{u_{ij}} \log_2 P_{ij}}. \quad (5.157)$$

□

Remark 5.5.1. In [68, Section 6.2.2], Soriaga considered the case where system requirements dictate that the expansion factor of the MDM code encoder cannot exceed f_0 , where $f_0 < f^*$. In such a case, the code cannot be an optimal MDM code for target probabilities $\{P_{ij}\}$. We can try to approximate an optimal MDM code by designing a code with $f \leq f_0$ that minimizes $D(f)$. Denote by $D_{\min}(f)$ the minimum achievable normalized KL-divergence when the expansion factor is upper bounded by f . By comparing equations (5.137) and (5.153), we have

$$\min_{\hat{P}_{u_{ij}}} \{D(f)\} = I_{\min}(f), \quad (5.158)$$

where $I_{\min}(f) = A_{\min}(f) - H(\mathbf{X})/f$, $A_{\min}(f)$ is the minimum average cost of a type-I shaping code with expansion factor f and edge costs $\{w(u_{ij}) = -\log_2 P_{ij}\}$ ($B^* = 1$). Since $I_{\min}(f)$ is monotone decreasing on $[H(\mathbf{X})/\log_2 \lambda(0), f^*)$, we can conclude that the minimum achievable

normalized KL-divergence when $f \leq f_0 < f^*$ is

$$D_{\min}(f_0) = I_{\min}(f_0) = \frac{H(\mathbf{X})}{B_0 f_0} - \frac{\log_2 \lambda(B_0)}{B_0} - \frac{H(\mathbf{X})}{f_0}, \quad (5.159)$$

where B_0 is associated with an optimal type-I shaping code with expansion factor f_0 . Based on the equivalence theorem we can extend the result in [68, Section 6.2.2] by concluding that this code is an optimal MDM code for transition probabilities $\{\hat{P}_{u_{ij}}^0 / \hat{P}_{s_i}^0\}$. $\square$

5.6 Generalized Expansion Factor

5.6.1 Motivation and Definition

The relationship between optimal shaping codes and optimal MDM codes was established above. The total cost of the shaping code suggests an alternative performance measure for a MDM code which will be useful when analyzing the optimality of a shaping-based MDM code construction and in proving a separation theorem for MDM codes. Specifically, we define the *generalized expansion factor* of a finite-state variable-length code as follows.

Definition 5.6.1. Given an irreducible graph $\mathcal{H}$, a set of positive real numbers $\{P_{ij}\}$ that are assigned to graph edges $\mathcal{U}$ that satisfy $\sum_j P_{ij} = 1$ for all i and a finite-state variable-length code $\phi : \mathcal{S} \times \mathcal{X}^q \rightarrow \mathcal{U}^*$ on the graph, the **generalized expansion factor** of this code is defined as

$$F(\phi, P_{ij}) = -f \frac{\sum_{ij} \hat{P}_{u_{ij}} \log_2 P_{ij}}{\log_2 \lambda(0)} \quad (5.160)$$

where f is the code's expansion factor, $\{\hat{P}_{u_{ij}}\}$ is the asymptotic symbol occurrence distribution of the codeword process, and $\lambda(0)$ is the Perron root of the adjacency matrix of graph $\mathcal{H}$. $\square$

For simplicity, we sometimes use F to represent $F(\phi, P_1, \dots, P_v)$. The following theorem shows that F can be used to evaluate an optimal MDM code.

Theorem 5.6.1. *Given a variable-length code $\phi: \mathcal{S} \times \mathcal{X}^q \rightarrow \mathcal{U}^*$ and a set of positive real numbers $\{P_{ij}\}$ such that $\sum_j P_{ij} = 1$ for all i , the generalized expansion factor of this mapping is lower bounded by*

$$F(\phi, P_{ij}) \geq \frac{H(\mathbf{X})}{\log_2 \lambda(0)}. \quad (5.161)$$

If $F = \frac{H(\mathbf{X})}{\log_2 \lambda(0)}$, this mapping is an optimal MDM code for transition probabilities $\{P_{ij}\}$. $\square$

Proof. Assume symbol u_{ij} in the codeword sequence has cost $\{w(u_{ij}) = -\log_2 P_{ij}\}$. The total cost of this mapping is

$$T(\phi) = f \sum_i \hat{P}_{u_{ij}} w(u_{ij}) = -f \sum_i \hat{P}_{u_{ij}} \log_2 P_{ij}. \quad (5.162)$$

Comparing equations (5.160) and (5.162), we have

$$F(\phi, P_{ij}) = \frac{T(\phi)}{\log_2 \lambda(0)}. \quad (5.163)$$

This indicates that GEF of a MDM code is equivalent to its total cost when applying it to a finite-state channel with edge costs $\{w(u_{ij}) = -\log_2 P_{ij}\}$. From Theorem 5.3.6, we know the total cost of a type-II shaping code satisfies the lower bound by

$$T(\phi) \geq \frac{H(\mathbf{X})}{B^*} \quad (5.164)$$

where B^* is a constant such that $\lambda(B^*) = 1$. Since $w(u_{ij}) = -\log_2 P_{ij}$, it is easy to check that $B^* = 1$ and

$$F(\phi, P_{ij}) = \frac{T(\phi)}{\log_2 \lambda(0)} \geq \frac{H(\mathbf{X})}{\log_2 \lambda(0)}. \quad (5.165)$$

When the minimum GEF is achieved, this code is also an optimal type-II shaping code. Theorem 5.5.2 then implies that this code is an optimal MDM code for the target transition probabilities $\{P_{ij}\}$. $\square$

Remark 5.6.1. As shown in Section 5.4.2, for a uniform i.i.d. source and a cost vector $\{w(u_{ij}) = -\log_2 P_{ij}\}$ that satisfied $\sum_j P_{ij} = 1$ for all i , a generalized Varn code $\phi : \mathcal{S} \times \mathcal{X}^q \rightarrow \mathcal{U}^*$ with codebook size M is an asymptotically optimal type-II shaping code and the total cost is bounded by

$$\begin{aligned} T(\phi) &\leq \frac{\log_2 M}{q} + \frac{\max_{ij}\{w'(u_{ij})\}}{q} \\ &\rightarrow \log_2 |\mathcal{X}| \quad \text{as } q \rightarrow \infty. \end{aligned} \quad (5.166)$$

Equation (5.163) then implies that for the target distribution $\{P_{ij}\}$, a generalized Varn code can be regarded as a MDM code with generalized expansion factor bounded by

$$\begin{aligned} F &\leq \frac{\log_2 M}{q \log_2 \lambda(0)} + \frac{\max_{ij}\{w'(u_{ij})\}}{q \log_2 \lambda(0)} \\ &\rightarrow \frac{\log_2 |\mathcal{X}|}{\log_2 \lambda(0)} \quad \text{as } q \rightarrow \infty. \end{aligned} \quad (5.167)$$

This implies that generalized Varn codes are asymptotically optimal MDM codes. $\square$

The separation theorem for shaping codes in Theorem 5.4.5 now extends naturally to MDM codes.

Theorem 5.6.2. *An optimal MDM code can be constructed by a concatenation of optimal lossless compression with an optimal MDM code for a uniform i.i.d. source, in the sense that the minimum generalized expansion factor can be achieved by such a concatenation.* $\square$

Remark 5.6.2. One special case is when P_{ij} can be represented by $\frac{v_j}{v_i \lambda(0)}$. As shown in Theorem 5.3.6, the generalized expansion factor reduces to

$$F = f \frac{\sum_{ij} \hat{P}_{u_{ij}} (\log_2 v_j - \log_2 v_i + \log_2 \lambda(0))}{\log_2 \lambda(0)} = f. \quad (5.168)$$

This provides the motivation for designating F by this name. $\square$

5.7 Acknowledgment

Chapter 5 is in part a reprint of the material in the paper: Yi Liu, Yonglong Li, Pengfei Huang, and Paul H. Siegel, Rate-Constrained Shaping Codes for Noiseless Finite-State Costly Channels, which is being prepared for submission to *IEEE Transactions on Information Theory*. The dissertation author was the primary investigator and author of this paper.

Chapter 6

Bad Page Detector for NAND Flash Memories

6.1 Introduction

In the previous chapters, we discussed shaping codes that are used to extend the lifetime of NAND flash memories. In this chapter, we take a different approach and develop a bad page detector. It predicts pages behavior in the near future based on the current and previous BECs. If a bad page is detected, it will be labeled and no future data will be written to it. The idea of bad page detection was mentioned in [50] and a support-vector machine (SVM) was used as the detector. In this chapter, we extend that work and propose two algorithms based on machine learning, including time-dependent neural network (TDNN) and long-short term memory (LSTM).

6.2 Bad Page Detector

In this section, we discuss the bad page detector for NAND flash memory. We first introduce the program and erase (P/E) cycling experiment, which is used to measure the performance

of TLC flash memory. We then define the training data set, which contains BEC information from all P/E cycle of a page. An overview of the bad page detection system is then presented.

6.2.1 P/E Cycling Experiment

The P/E cycling experiment consists of repeating the following steps which collectively represent one P/E cycle:

- Erase the block(s).
- Program the pages with pseudo-random data.
- Starting with the first cycle, perform a read operation on the the block(s) at intervals of every 100 cycle. Record bit errors, their locations in the block(s).

The TLC flash memory blocks are P/E cycled up to 10,000 P/E cycles and the P/E cycling experiments are performed at room temperature in a continuous manner with no wait time between the erase/program/read operations. Figs. 6.1, 6.2 and 6.3 show the behavior of all pages been measured in our experiment. Each line represents the BEC vs. P/E cycle count of one page. As shown in the figures, the behavior of pages varies significantly. The data collected consists of sequences of pages BECs, $[N(T_1), N(T_2), \dots, N(T_l)]$, associated with P/E cycle counts $[T_1, T_2, \dots, T_l]$.

6.2.2 $(T, T_{\text{Of}}, N_{\text{Th}})$ —Bad Page

We formally define a $(T, T_{\text{Of}}, N_{\text{Th}})$ -bad page as follows.

Definition 6.2.1. In a flash memory device, a certain physical page is called a $(T, T_{\text{Of}}, N_{\text{Th}})$ -bad page if its BEC at some P/E cycle count less than $T + T_{\text{Of}}$ is not less than N_{Th} .

For example, a $(3000, 500, 1200)$ -bad page has a BEC at some P/E cycle count before 3500 that is greater than 1200. Notice that this page is also a $(T, 500, 1200)$ -bad page for any

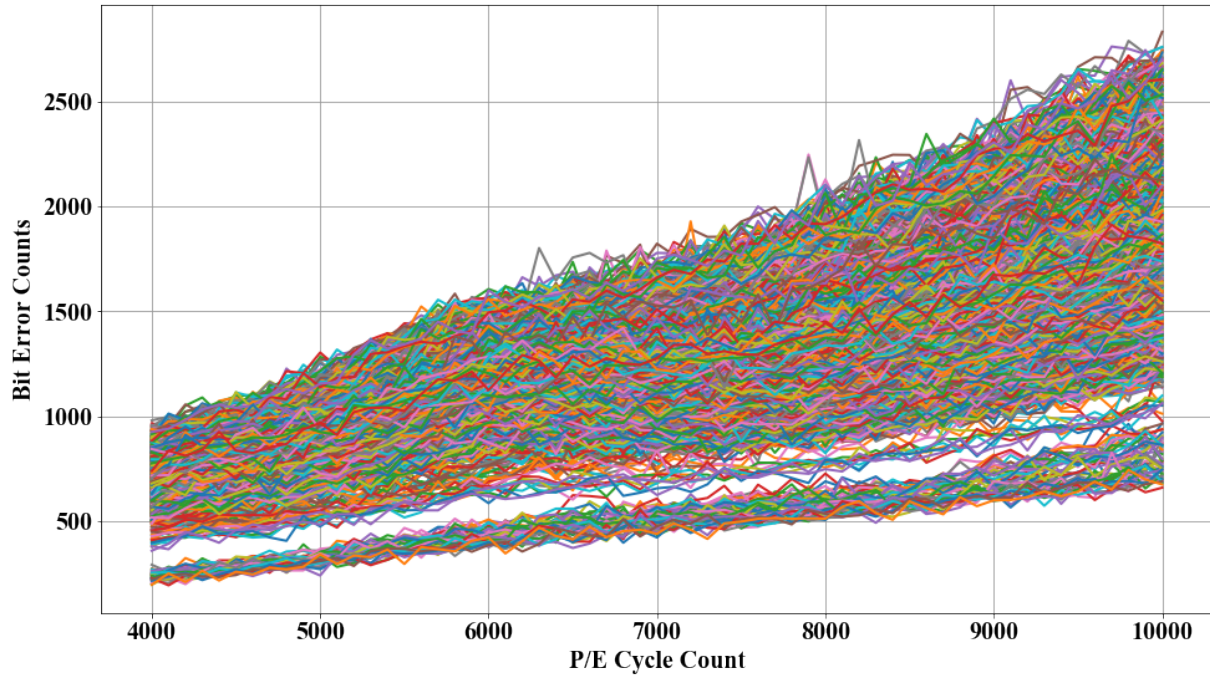


Figure 6.1: Measured bit error counts vs. P/E cycle count of lower pages.

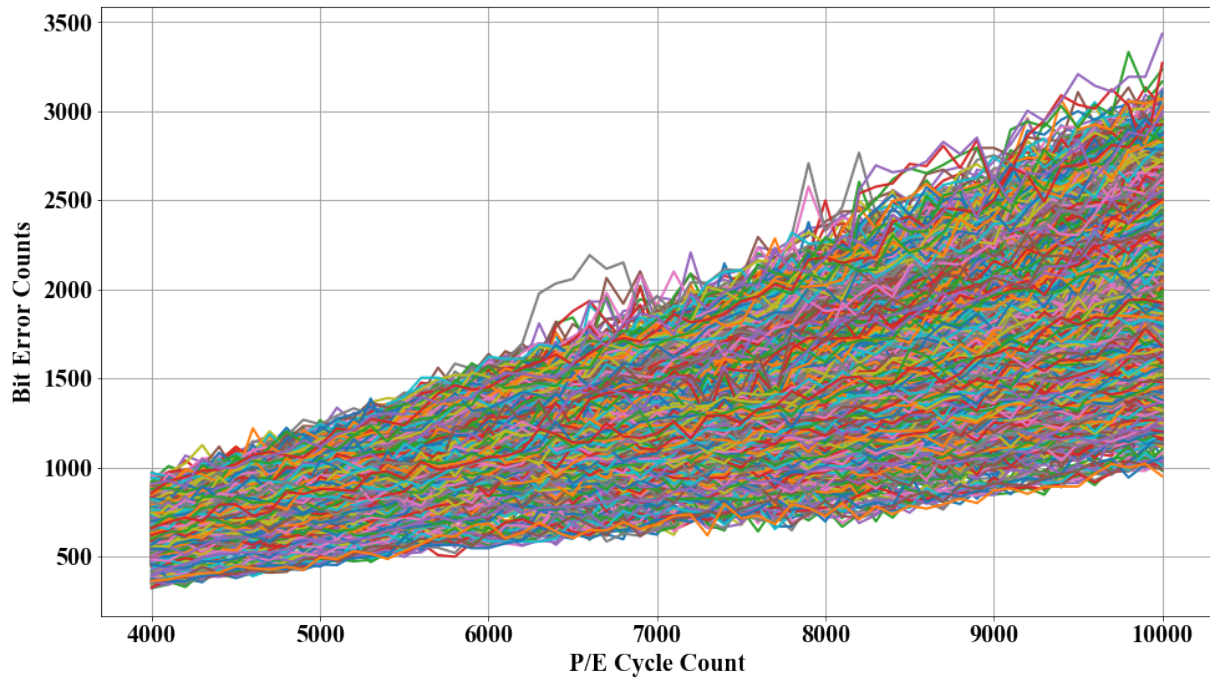


Figure 6.2: Measured bit error counts vs. P/E cycle count of middle pages.

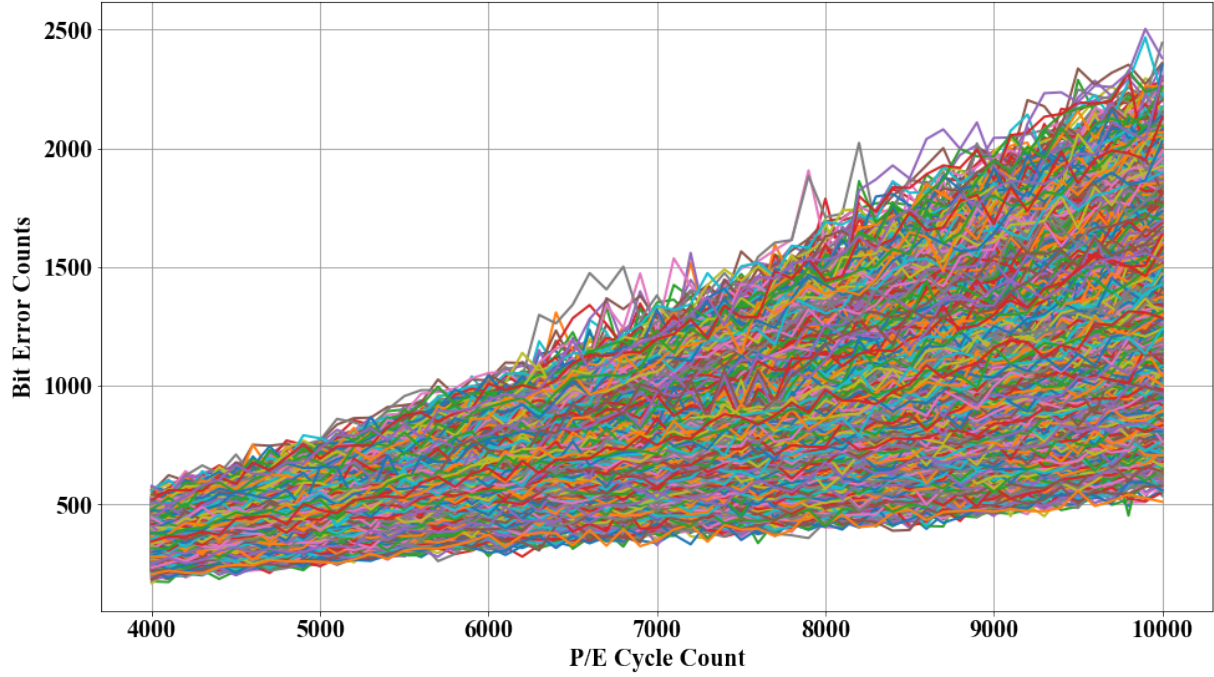


Figure 6.3: Measured bit error counts vs. P/E cycle count of upper pages.

$T \geq 3000$. Based on this definition, we introduce the labeling function, which has T , T_{Of} , N_{Th} and a BEC sequence $N(T)$ as input, and is defined as follows.

$$L(T, T_{\text{Of}}, N, N_{\text{Th}}) = \begin{cases} 1, & \text{if } N(T') \geq N_{\text{Th}}, T' \leq T + T_{\text{Of}} \\ 0, & \text{otherwise} \end{cases} \quad (6.1)$$

The output 1 indicates that this page is a $(T, T_{\text{Of}}, N_{\text{Th}})$ -bad page. An example is shown schematically in Fig. 6.4.

In this chapter, we train the detector to predict $(T, 500, 1200)$ -bad page by making use of the page's information before T . The input is the vector of BECs $[N(T_1), N(T_2), \dots, N(T_5)]$, where $T_{i+1} - T_i = 100$ and $T_5 = T$. The training dataset $\mathcal{X}$ consists of pairs $(\mathbf{x}, t)$, where $\mathbf{x}$ is a length-5 vector,

$$\begin{aligned} \mathbf{x} &= [N(T_1), N(T_2), \dots, N(T_5)], \\ T_{i+1} - T_i &= 100 \quad \text{and} \quad 4,000 \leq T_1 < T_5 \leq 10,000, \end{aligned} \quad (6.2)$$

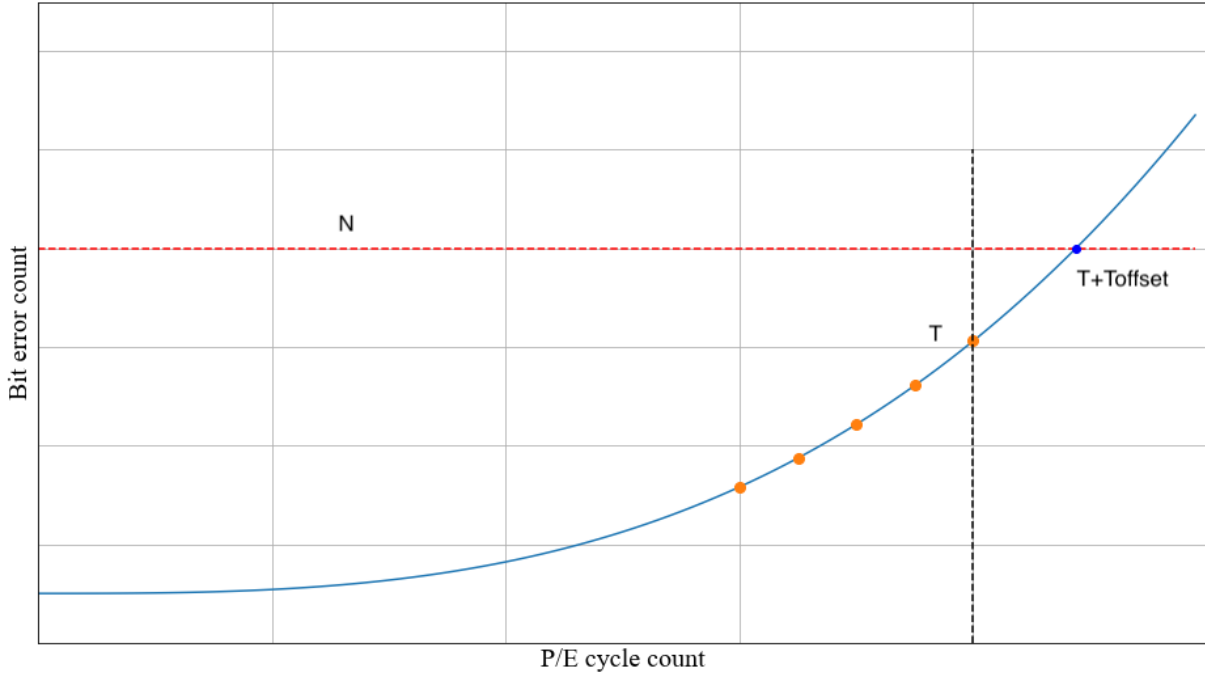


Figure 6.4: Example of an $(T, T_{\text{Of}}, N_{\text{Th}})$ -bad page.

and t is defined by resizing T_5

$$t = \frac{T_5 - 4000}{10000 - 4000} \quad (6.3)$$

such that t ranges between $[0, 1]$. The P/E cycling experiment collects data from 12,855 pages, so the dataset size is $12,855 \times 52 = 668,460$ (from P/E cycle count 4,000 – 4,400 to 9,100 – 9,500).

In [50], a bad page detector based on a support vector machine was proposed. The detector is trained on a subset of $\mathcal{X}$ with fixed t . Given the BEC vector $\mathbf{x}$, the detector calculates the following function

$$f(\mathbf{x}) = \sum w_i x_i + b. \quad (6.4)$$

If $f(\mathbf{x}) < 0$, the page is labeled as a good page and if $f(\mathbf{x}) \geq 0$, this page is labeled as a bad page. The trade-off between accuracy and recall was also studied. The authors showed that the detector can detect all bad pages (100% recall) and the overall accuracy is 97.6%. Even though the detector is only trained at a fixed P/E cycle count, it also works well at other P/E cycle counts.

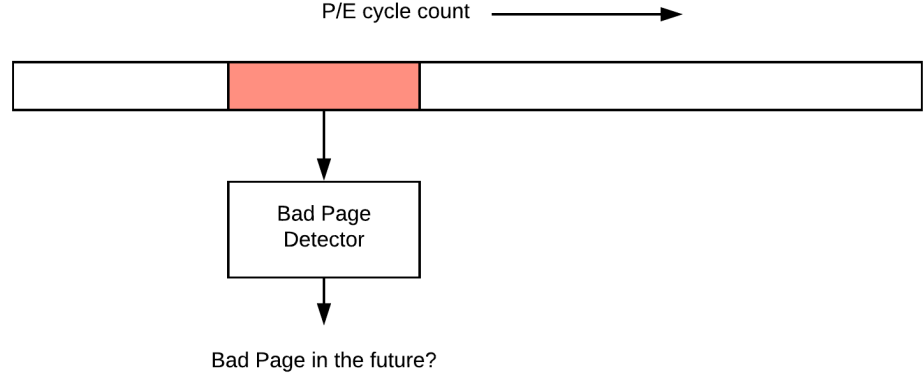


Figure 6.5: Bad Page Detection System.

6.2.3 Bad Page Detector Overview

Given a sequence of BECs $[N(T_1), N(T_2), \dots, N(T_l)]$, the bad page detector predicts its performance using a length-5 sliding window. For example, at P/E cycle count T_4 , $[N(T_i), \dots, N(T_{i+4})]$ is fed into the detector. If the detector determines that this page is a $(T_4, 500, 1200)$ -bad page, no future data will be written to it. If not, we keep on using this page until T_5 ($T_5 - T_4 = 100$) and do the test again. The procedure is illustrated in Fig. 6.5.

6.3 Time-Dependent Neural Network

In this section, we discuss the structure of a time dependent neural network (TDNN), whose weights and biases are functions of t . Its motivation comes from the fact that BEC behavior varies as a function of P/E cycle count. In [29], this network was used to simulate character walking.

6.3.1 Network Structure

TDNN consists of three kinds of layers: time-dependent dense layer $\mathcal{D}(\mathbf{x}, t)$ (TDDL), leaky ReLU layer $\mathcal{L}(\mathbf{x})$, and softmax layer $\mathcal{S}(\mathbf{x})$. The details of these layers are listed below.

- Time-dependent dense layer $\mathcal{D}(\mathbf{x}, t)$ (TDDL) with input size m and output size n : given an input vector $\mathbf{x} \in \mathbb{R}^m$, TDDL defines the following operation

$$\mathbf{y} = \mathcal{D}(\mathbf{x}, t) = \mathbf{W}(t)\mathbf{x} + \mathbf{B}(t) \quad (6.5)$$

where $\mathbf{y} \in \mathbb{R}^n$ is the output vector and $\mathbf{W}(t) = \{w_{ij}(t)\} \in \mathbb{R}^{n \times m}$, $\mathbf{B}(t) = \{B_i(t)\}^\top \in \mathbb{R}^n$ are functions of P/E cycle count to be determined during the training process. If the functions are constant, the layer is a dense layer, denoted simply by $\mathcal{D}(\mathbf{x})$.

- Leaky rectifier linear unit (leaky ReLU) $\mathbf{y} = \mathcal{L}(\mathbf{x})$: given a input vector $\mathbf{x} \in \mathbb{R}^m$, the output of the leaky ReLU layer $\mathbf{y} \in \mathbb{R}^m$ is

$$\mathbf{y} = \mathcal{L}(\mathbf{x}) : y_i = \begin{cases} x_i, & \text{if } x_i \geq 0 \\ 0.01x_i, & \text{otherwise.} \end{cases} \quad (6.6)$$

- Softmax layer $\mathbf{y} = \mathcal{S}(\mathbf{x})$: given an input vector $\mathbf{x} \in \mathbb{R}^m$, the output of softmax layer $\mathbf{y} \in \mathbb{R}^m$ is

$$\mathbf{y} = \mathcal{S}(\mathbf{x}) : y_i = \frac{e^{x_i}}{\sum_{i=1}^m e^{x_i}}. \quad (6.7)$$

The TDNN $\Phi(\mathbf{x}, t)$ contains two TDDLs $\mathcal{D}_0(\mathbf{x}, t)$ and $\mathcal{D}_1(\mathbf{x}, t)$ with input size 5 and output size 5; one dense layer $\mathcal{D}_2(\mathbf{x})$ with input size 5 and output size 2; two leaky ReLU layers $\mathcal{L}$; and one softmax layer $\mathcal{S}$. We build the network $\Phi(\mathbf{x}, t)$ as follows:

$$\mathbf{y} = \Phi(\mathbf{x}, t) = \mathcal{S}(\mathcal{D}_2(\mathcal{L}(\mathcal{D}_1(\mathcal{L}(\mathcal{D}_0(\mathbf{x}, t))), t))). \quad (6.8)$$

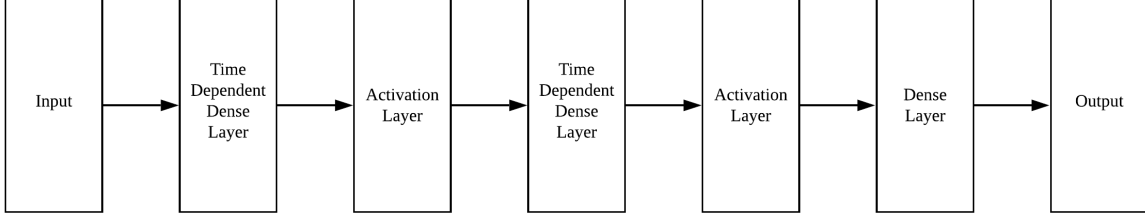


Figure 6.6: The structure of TDNN.

The network structure is shown in Fig. 6.6.

6.3.2 Approximating the Time Dependent Dense Layer

Weight and bias functions in a TDDL can be approximated by degree- n polynomial functions $f(t) = \sum_{k=0}^n a_k t^k$, where $\{a_i\}$ are the control parameters that are determined during the training process. The weight matrix $\mathbf{W}(t)$ in a TDDL can be represented as

$$\begin{aligned}
 \mathbf{W}(t) &= \{w_{ij}(t)\} = \left\{ \sum_{k=0}^n a_k^{ij} t^k \right\} \\
 &= \sum_{k=0}^n \{a_k^{ij}\} t^k = \sum_{k=0}^n \mathbf{W}_k t^k,
 \end{aligned} \tag{6.9}$$

in which $\mathbf{W}_k = \{a_k^{ij}\}$ is a constant matrix. Therefore $\mathcal{D}(\mathbf{x}, t)$ can be represented by

$$\mathcal{D}(\mathbf{x}, t) = \sum_{k=0}^n \mathcal{D}_k(\mathbf{x}) t^k. \tag{6.10}$$

This indicates that a degree- n TDDL can be realized by connecting $n + 1$ dense layers in parallel, multiplied by t_i , $0 \leq i \leq n$, respectively. An example of a degree-3 TDDL is shown in Fig. 6.7. In general, higher degree polynomials would result in a more accurate approximation.

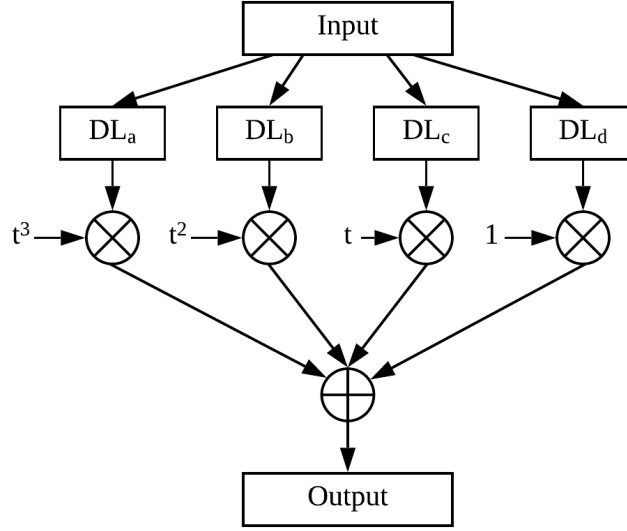


Figure 6.7: The realization of time dependent dense layer. DLs represent dense layers.

6.3.3 Training the Network

The output of TDNN $\mathbf{y} = [p_1, p_2]$ represents the probability of being a bad page ($p_1 + p_2 = 1$). We predict the page is a bad page if, for a specified threshold probability p_{Th} (usually set to 0.5), $p_1 \geq p_{Th}$. The loss function used in the training process is cross entropy. Given a batch of input data $[(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2), \dots, (\mathbf{x}_s, t_s)]$ and labels $[l_1, l_2, \dots, l_s]$, denote by $[p_1^i, p_2^i] = \Phi(\mathbf{x}_i, t_i)$, we solve the optimization problem that minimizes the following loss function

$$C = -\frac{1}{s} \sum_i E(l_i) [q_1(l_i) \log p_1^i + q_2(l_i) \log p_2^i], \quad (6.11)$$

where $q_j(l)$ is the label ($q_1(1) = q_2(0) = 1$, $q_1(0) = q_2(1) = 0$). $E(l_i)$ is the loss function weights

$$E(l_i) = \begin{cases} 1.3865 & \text{if } l_i = 1 \\ 1 & \text{if } l_i = 0. \end{cases} \quad (6.12)$$

We train the detector using Tensorflow. About 90 percent of the data is used as the training

data and 10 percent is used as the testing data. The training is performed with batch size $s = 2000$, learning rate 0.001 and for 400000 iterations.

6.4 Long-Short Term Memory

The LSTM detector predicts bad pages by solving a regression problem. The network takes a sequence of l —consecutive BECs and predicts the the BEC at a future P/E cycle count. The labeling function L_{LSTM} is given by

$$L_{\text{LSTM}}(\mathbf{x}) = N(T_5 + 500) \quad (6.13)$$

for any $\mathbf{x} \in \mathcal{X}$. The LSTM network consists of a standard LSTM layer [28] $\mathcal{M} : \mathbb{R}^5 \rightarrow \mathbb{R}^4$ and a dense layer $\mathcal{D} : \mathbb{R}^4 \rightarrow \mathbb{R}$, represented as $y = \mathcal{D}(\mathcal{M}(\mathbf{x}))$. We predict the page is a bad page if, for a specified threshold y_{Th} (usually set to N_{Th}), $y \geq y_{\text{Th}}$.

6.5 Experiment Results

In this section, we present experiment results of different bad page detectors. Since the bad page detector is a “sliding window” detector, simply looking at the accuracy or recall of the training result is not enough to measure the performance of the detector. We first define Q -bad page, which is the difference between P/E cycle counts when the page is detected as bad page and when the page actually become a bad page. We then compare performances between different bad page detectors by looking at the distribution of Q -bad pages.

6.5.1 Q -Bad Page

Given a page and its BEC vector $N(T)$, we denote by T_l the first time the page is labeled as a bad page, which means $L(T_l, T_{\text{Of}}, N, N_{\text{Th}}) = 1$ and $L(T, T_{\text{Of}}, N, N_{\text{Th}}) = 0$ for any $T < T_l$.

Similarly, we denote by T_d the first time the page is classified as a bad page by the detector. Letting $Q = T_d - T_l$, we say this page is a Q -bad page. All pages in the dataset can be categorized into three groups based on Q .

- Group-I: when $Q < 0$, the page is labeled before it actually becomes a bad page. Its storage capacity is wasted.
- Group-II: when $0 < Q < 500$, even though the detector fails to label this page correctly at $T_d - 500$, the offset T_{Of} ensures that the detector still labels it bad *before* its BEC actually surpasses N_{Th} .
- Group-III: when $Q \geq 500$, the detector fails to predict the bad page before it has become bad. These pages are also referred to as mispredicted pages.

The goal of the bad page detector is to reduce the number of mispredicted bad pages (Group III) while also controlling the size of wasted P/E cycles (Group I).

Remark 6.5.1. By adjusting the detection threshold (P_{Th} for TDNN and y_{Th} for LSTM), we are able to modify the ratio of each group. For example, if P_{Th} decreases, the detector will predict a page as a bad page at an earlier P/E cycle count. Therefore the size of group-III will decrease and the size of group-I will increase. In other words, to reduce the total number of late prediction (high recall), more storage capacity is wasted.

6.5.2 Analysis of Results

Experimental results are shown in Fig. 6.8 and Table 6.1. To compare the performances, the detecting thresholds were tuned such that the numbers of the mispredicted pages in different detectors are close to each other. We can draw the following conclusions.

- TDNN with higher degree has better performance.
- TDNNs with input size 5 and 7 outperform TDNN with input size 3.

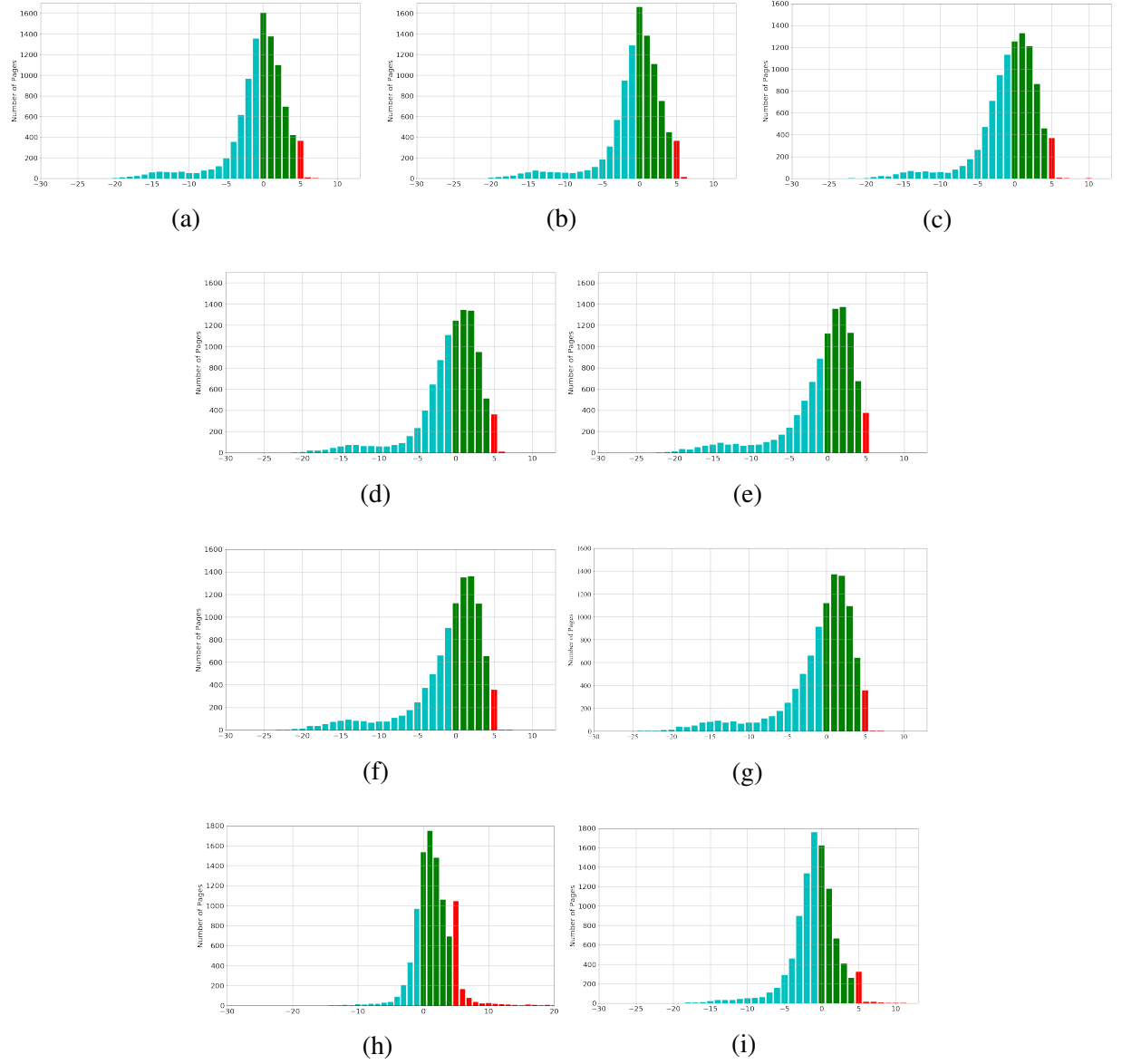
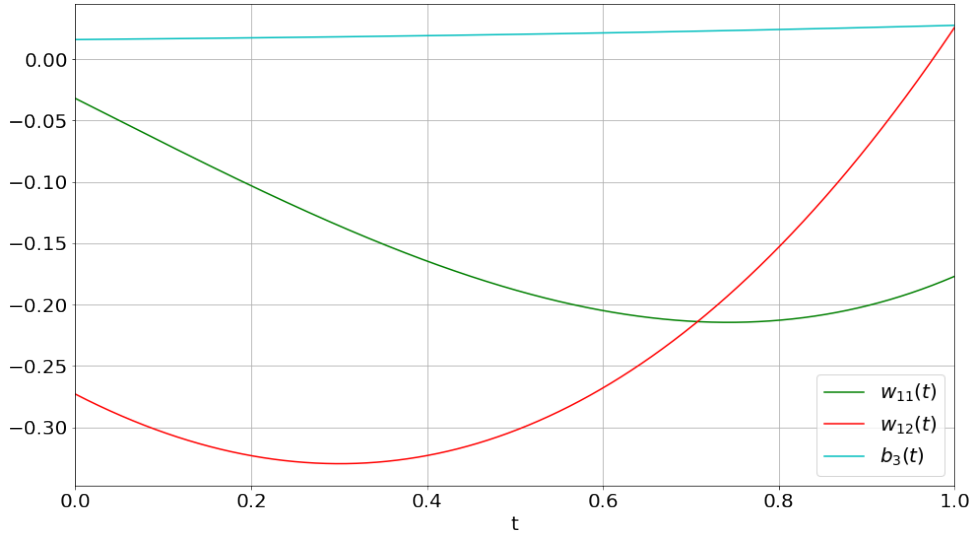


Figure 6.8: Distribution of Q -bad pages under different input vector-length and network structures. (a) Degree-3 TDNN with input length 5. (b) Degree-3 TDNN with input length 7. (c) Degree-3 TDNN with input length 3. (d) Linear TDNN with input length 5. (e) Constant TDNN with input length 5. (f) Constant TDNN with input length 5. (g) SVM with input length 5. (h) LSTM with input length 5. (i) LSTM with input length 5 and $y_{Th} = 1163$.

- TDNN and LSTM have similar performance.

Table 6.1: Performance of TDNN, LSTM and SVM.

Network	Input length	Degree	Phase function	Number of mispredicted pages	wasted P/E cycles($\times 100$)
Cubic TDNN with input length 5	5	3	$f(t) = at^3 + bt^3 + ct + d$	375	15982
Cubic TDNN with input length 7	7	3	$f(t) = at^3 + bt^3 + ct + d$	378	16012
Cubic TDNN with input length 3	3	3	$f(t) = at^3 + bt^3 + ct + d$	378	17436
Linear TDNN with input length 5	5	1	$f(t) = at + b(1-t)$	375	17196
Constant TDNN with input length 5	5	0	$f(t) = a$	376	19064
Constant TDNN with input length 5	5	0	$f(t) = a$	358	19519
Support vector machine with input length 5	5	0	$f(t) = a$	358	19761
Cubic TDNN with input length 5	5	3	$f(t) = at^3 + bt^3 + ct + d$	72	37830
LSTM with $y_{Th} = 1200$	5			1449	3888
LSTM with $y_{Th} = 1163$	5			376	16084

**Figure 6.9:** Functions in the cubic TDNN with input length 5.

We also looked at the trained functions in the cubic TDNN with input length 5 (Fig. 6.8a). The weight function $w_{11}(t)$ in $\mathfrak{D}_0(\mathbf{x}, t)$ is

$$w_{11}(t) = 0.236t^3 - 0.020t^2 - 0.361t - 0.032. \quad (6.14)$$

This function, together with $w_{12}(t)$ and $b_3(t)$, are shown in Fig. 6.9.

As mentioned in Remark 6.5.1, in real applications, sometimes we want the detector to be able to detect almost all bad pages before their BECs surpass the threshold (high recall). This can be done by decreasing the detecting threshold p_{Th} . Fig. 6.10a shows the distribution

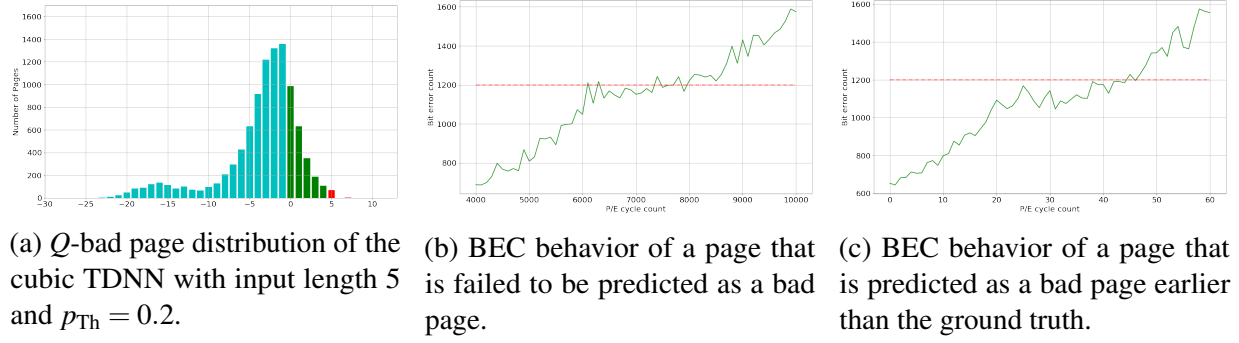


Figure 6.10: Error analysis

of Q -bad pages of a cubic TDNN with input length 5 when setting $p_{Th} = 0.2$. Compared to Fig. 6.8a, the total number of mispredicted pages decreases from 375 to 72. However the total number of wasted P/E cycle counts increases from 15982 to 37830. There is a trade-off between mispredicted pages and wasted P/E cycle counts. In order for a detector to be able to detect almost all bad pages, the wasted P/E cycle counts will inevitably be large.

Another observation from Fig. 6.10a is that some pages still cannot be detected. The BEC behavior of one of those pages is shown in Fig. 6.10b. We can see from the figure that its BEC increases rapidly from 6000 to 6100 and the detector cannot predict its behavior. There are also pages that are predicted as bad pages too early ($Q \leq -1500$). The BEC behavior of one of those pages is shown in Fig. 6.10c. Its BEC oscillates between 1000 to 1200. This results in an early prediction.

6.6 Acknowledgment

Chapter 6 is in part a reprint of the material in the paper: Yi Liu, Si Wu, and Paul H. Siegel, Bad Page Detector for NAND Flash Memory, in *Proc. 11th Annual Non-Volatile Memories Workshop (NVMW)*, La Jolla, CA, USA, Mar. 2020. The dissertation author was the primary investigator and author of this paper.

Bibliography

- [1] J. Abrahams, “Variable-length unequal cost parsing and coding for shaping,” *IEEE Trans. Inf. Theory*, vol. 44, no. 4, pp. 1648–1649, Jul. 1998.
- [2] R. Adler, D. Coppersmith, and M. Hassner, “Algorithms for sliding block codes,” *IEEE Trans. Inf. Theory*, vol. IT-29, no. 1, pp. 5–22, Jan. 1983.
- [3] R. A. Amjad and G. Böcherer, “Fixed-to-variable length distribution matching,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Istanbul, Turkey, Jul. 2013, pp. 1511–1515.
- [4] S. Baur and G. Bcherer, “Arithmetic distribution matching,” in *Proc. Int. ITG Conf. Source Channel Coding (SCC)*, Hamburg, Germany, Feb. 2-5, 2015, pp. 1–6.
- [5] G. Böcherer, “Capacity-achieving probabilistic shaping for noisy and noiseless channels,” PhD thesis, RWTH Aachen University, 2012. [Online]. Available: <http://www.georg-boecherer.de/capacityAchievingShaping.pdf>.
- [6] G. Böcherer and R. A. Amjad, “Informational divergence and entropy rate on rooted trees with probabilities,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, 2014, pp. 176–180.
- [7] G. Böcherer and R. Mathar, “LDPC codes with zero shaping gap,” in *Proc. IEEE Inf. Theory Workshop (ITW)*, 2011.
- [8] G. Böcherer and R. Mathar, “Matching dyadic distributions to channels,” in *Proc. Data Compression Conf. (DCC)*, 2011, pp. 23–32.
- [9] G. Böcherer, V. C. da Rocha Jr., C. Pimentel, and R. Mathar, “On the capacity of constrained systems,” in *Proc. Int. ITG Conf. Source Channel Coding (SCC)*, 2010.
- [10] G. Böcherer, F. Steiner, and P. Schulte, “Bandwidth efficient and rate-matched low-density parity-check coded modulation,” *IEEE Trans. Commun.*, vol. 63, no. 12, pp. 4651–4665, Dec. 2015.
- [11] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.
- [12] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd ed. Hoboken, NJ, USA: Wiley, 2006.

- [13] I. Csiszár, “Simple proofs of some theorems on noiseless channels,” *Inf. Contr.*, vol. 14, pp. 285–298, 1969.
- [14] I. Csiszár and J. Körner, *Information Theory: Coding Theorems for Discrete Memoryless Systems*. Cambridge, U.K.: Cambridge University Press, 2011.
- [15] R. Durrett, *Probability: Theory and Examples*, 5th ed. Cambridge university press, 2019.
- [16] R. F. H. Fischer, *Precoding and Signal Shaping for Digital Transmission*. Hoboken, NJ, USA: Wiley, 2002.
- [17] G. Forney, R. Gallager, G. Lang, F. Longstaff and S. Qureshi, “Efficient modulation for band-limited channels,” *IEEE J. Select. Areas Commun.*, vol. 2, no. 5, pp. 632–647, Sep. 1984.
- [18] R. Fujita, K. Iwata, and H. Yamamoto, “An Iterative Algorithm to Optimize the Average Performance of Markov Chains with Finite States,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Paris, France, 2019, pp.1902 - 1906.
- [19] R. G. Gallager, *Information Theory and Reliable Communication*. New York, NY, USA: Wiley, 1968.
- [20] E. N. Gilbert, “Coding with digits of unequal cost,” *IEEE Trans. Inf. Theory*, vol. 41, no. 2, pp. 596–600, Mar. 1995.
- [21] M. J. Golin and G. Rote, “A dynamic programming algorithm for constructing optimal prefix-free codes with unequal letter costs,” *IEEE Trans. Inf. Theory*, vol. 44, no. 5, pp. 1770–1781, Sep. 1998.
- [22] M. Goto, T. Matsushima, and S. Hirasawa, “A Source Model with Probability Distribution over Word Set and Recurrence Time Theorem”, *IEICE Trans. Fundamentals*, vol. E86-A, no. 10, Oct. 2003.
- [23] M. Guazzo, “A general minimum-redundancy source-coding algorithm,” *IEEE Trans. Inf. Theory*, vol. 26, no. 1, pp. 15–25, Jan. 1980.
- [24] Y. C. Gültekin, W. J. van Houtum, S. Şerbetli, and F. M. Willems, “Constellation shaping for IEEE 802.11,” in *Proc. IEEE Int. Symp. Personal, Indoor, Mobile Radio Commun. (PIMRC)*, 2017.
- [25] T.S. Han, *Information-Spectrum Methods in Information Theory*. Berlin, Germany: Springer-Verlag, 2003, (originally published by Baifukan 1998 in Japanese).
- [26] T.S. Han and O. Uchida, “Source code with cost as a nonuniform random number generator,” *IEEE Trans. Inf. Theory*, vol. 46, no. 2, pp. 712–717, Mar. 2000.

- [27] C. D. Heegard, B. H. Marcus, and P. H. Siegel, "Variable-length state splitting with applications to average runlength-constrained (ARC) codes," *IEEE Trans. Inf. Theory*, vol. 37, no. 3, pp. 759–777, May 1991.
- [28] S. Hochreiter, and J. Schmidhuber, "Long short-term memory", *Neural Computation*, 9(8):1735-1780, 1997.
- [29] D. Holden, T. Komura, and J. Saitou, "Phase-functioned neural networks for character control," *ACM Trans. Graphics (TOG)*, volume 36, issue 4, Jul. 2017.
- [30] K. Iwata, M. Morii, and T. Uyematsu, "An efficient universal coding algorithm for noiseless channel with symbols of unequal cost," *IEICE Trans. Fundamentals*, vol. E80-A, no. 11, pp. 2232–2237, Nov. 1997.
- [31] A. Jagmohan, M. Franceschini, L. A. Lastras-Montaña and J. Karidis, "Adaptive endurance coding for NAND Flash," in *Proc. IEEE GLOBECOM Workshops*, Dec. 2010, pp. 1841–1845.
- [32] J. Justesen and T. Høholdt, "Maxentropic Markov chains," *IEEE Trans. Inf. Theory*, vol. IT-30, no. 4, pp. 665–667, Jul. 1984.
- [33] R. Karabed, D. L. Neuhoff, A. Khayrallah, *The Capacity of Costly Noiseless Channels*, Research report, IBM Research Division, 1988.
- [34] R. Karp, "Minimum-redundancy coding for the discrete noiseless channel," *IRE IEEE Trans. Inf. Theory*, vol. 7, no. 1, pp. 27–38, Jan. 1961.
- [35] A. Khandekar, R. J. McEliece, and E. Rodemich, "The discrete noiseless channel revisited," in *Proc. 1999 Int. Symp. Communication Theory and Applications*, pp. 115–137, 1999.
- [36] A. S. Khayrallah and D. L. Neuhoff, "Coding for channels with cost constraints," *IEEE Trans. Inf. Theory*, vol. 42, no. 3, pp. 854–867, May 1996.
- [37] V. Y. Krachkovsky, R. Karabed, S. Yang, and B. A. Wilson, "On modulation coding for channels with cost constraints," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Honolulu, Hawaii, Jun. 29-Jul. 4, 2014, pp. 421–425.
- [38] R. M. Krause, "Channels which transmit letters of unequal duration," *Inf. Contr.*, vol. 55, pp. 13–24, 1962.
- [39] F. R. Kschischang and S. Pasupathy, "Optimal nonuniform signaling for Gaussian channels," *IEEE Trans. Inf. Theory*, vol. 39, no. 3, pp. 913–929, May 1993. May 1993.
- [40] A. Lempel, S. Even, and M. Cohn, "An algorithm for optimal prefix parsing of a noiseless and memoryless channel," *IEEE Trans. Inf. Theory*, vol. 19, no. 2, pp. 208–214, Mar. 1973.
- [41] A. Lenz, Y. Liu, C. Rashtchian, P. H. Siegel, A. Wachter-Zeh, and E. Yaakobi, "Coding for Efficient DNA Synthesis," submitted to *IEEE Int. Symp. Inf. Theory*, 2020.

- [42] J. Li, K. Zhao, X. Zhang, J. Ma, M. Zhao, and T. Zhang, “How much can data compressibility help to improve NAND flash memory lifetime?,” in *Proc. 13th USENIX Conf. File and Storage Technologies (FAST’15)*, USENIX Assoc., Berkeley, CA, Feb. 16-19, 2015, pp. 227-240.
- [43] Y. Liu and P. H. Siegel, “Shaping codes for structured data,” *7th Annual Non-Volatile Memories Workshop (NVMW)*, La Jolla, CA, Mar. 6–8, 2016.
- [44] Y. Liu and P. H. Siegel, “Shaping codes for structured data,” in *Proc. IEEE Globecom*, Washington, D.C., Dec. 4-8, 2016, pp. 1–5.
- [45] Y. Liu, P. Huang and P. H. Siegel, “Performance of shaping codes for flash memory,” *8th Annual Non-Volatile Memories Workshop (NVMW)*, La Jolla, CA, Mar. 12–14, 2017.
- [46] Y. Liu, P. Huang and P. H. Siegel, “Performance of optimal data shaping codes,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Aachen, Germany, Jun. 25-30, 2017, pp. 1003–1007.
- [47] Y. Liu, P. Huang, A. W. Bergman and P. H. Siegel, “Optimal data shaping code design,” *9th Annual Non-Volatile Memories Workshop (NVMW)*, La Jolla, CA, Mar. 11-13, 2018.
- [48] Y. Liu, P. Huang, A. W. Bergman, P. H. Siegel, “Rate-Constrained Shaping Codes for Structured Sources”, submitted to *IEEE Trans. Inf. Theory*.
- [49] R. S. Marcus, “Discrete noiseless coding,” Master’s thesis, Massachusetts Institute of Technology, 1957.
- [50] Y. Liu, and X. Zhang, “Using machine learning algorithms to map out bad pages in NAND flash memory”, presented at 2018 CMRR Spring Research Review, La Jolla, CA, Mar 15, 2018.
- [51] R. J. McEliece, *The Theory of Information and Coding*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 2001.
- [52] R. J. McEliece and E. R. Rodemich, “A maximum entropy Markov chain,” in *Proc. 17th Conf. Inf. Sciences and Systems*, Johns Hopkins University, Mar. 1983, pp. 245–248.
- [53] S. W. McLaughlin and A. S. Khayrallah, “Shaping codes constructed from cost-constrained graphs,” *IEEE Trans. Inf. Theory*, vol. 43, no. 2, pp. 692–699, Mar. 1997.
- [54] K. Mehlhorn, “An efficient algorithm for constructing nearly optimal prefix codes,” *IEEE Trans. Inf. Theory*, vol. 26, no. 5, pp. 513–517, Sep. 1980.
- [55] M. Mondelli, S. H. Hassani, and R. L. Urbanke, “How to achieve the capacity of asymmetric channels,” *IEEE Trans. Inf. Theory*, vol. 64, no. 5, pp. 3371–3393, 2018.
- [56] M. Nishiara and H. Morita, “On the AEP of word-valued sources,” *IEEE Trans. Inf. Theory*, vol. 46, no. 3, pp. 1116–1120, May 2000.

- [57] R. D. Nussbaum, “Convexity and Log Convexity for the Spectral Radius,” *Linear Algebra and its Applications*, 73:59-122, 1986.
- [58] T. V. Ramabadran, “A coding scheme for m-out-of-n codes,” *IEEE Trans. Commun.*, vol. 38, no. 8, pp. 1156–1163, Aug. 1990.
- [59] RANDOM.ORG, 2018. [online]. Available: <https://www.random.org>.
- [60] A. Rukhin, J. Soto, J. Nachvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, J. Dray, and S. Vo, “A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications,” NIST Special Publication 800-22 Revision 1a, Apr. 2010.
- [61] S. A. Savari, “Some notes on Varn coding,” *IEEE Trans. Inf. Theory*, vol. 40, no. 1, pp. 181–186, Jan. 1994.
- [62] S. A. Savari and R. G. Gallager, “Arithmetic coding for finite-state noiseless channels,” *IEEE Trans. Inf. Theory*, vol. 40, no. 1, pp. 100–107, Jan. 1994.
- [63] S. A. Savari and R. G. Gallager, “Generalized Tunstall Codes for Sources with Memory,” *IEEE Trans. Inf. Theory*, vol. 43, no. 2, pp. 658-668, 1997.
- [64] P. Schulte and G. Böcherer, “Constant composition distribution matching,” *IEEE Trans. Inf. Theory*, vol. 62, no. 1, pp. 430–434, Jan. 2016.
- [65] P. Schulte and F. Steiner, “Divergence-optimal fixed-to-fixed length distribution matching with shell mapping,” *IEEE Wireless Commun. Letters*, vol.8, no.2, pp. 620-623, Apr. 2019.
- [66] C. E. Shannon, “A mathematical theory of communication, Part I, Part II,” *Bell Syst. Tech. J.*, vol. 27, pp. 379–423, 1948.
- [67] E. Sharon, S. Achtenberg, I. Alrod, A. Klein, and A. Eyal, “Data Shaping for Improving Endurance and Reliability in Sub-20nm NAND,” presented at *Flash Memory Summit*, Santa Clara, CA, Aug. 4–7, 2014.
- [68] J. Soriaga, *On Near-Capacity Code Design for Partial-Response Channels*. Ph.D. thesis, University of California, San Diego, La Jolla, CA, USA, Mar. 2005.
- [69] J. B. Soriaga and P. H. Siegel, “Near-capacity coding systems for partial-response channels,” in *Proc. IEEE Int. Symp. Inf. Theory*, Chicago, IL, USA, p. 267, June 2004.
- [70] J. B. Soriaga and P. H. Siegel “On the Design of Finite-State Shaping Encoders for Partial-Response Channels,” in *Proc. of the 2006 Inf. Theory and Application Workshop (ITA2006)*, San Diego, CA, USA, Feb. 2006.
- [71] S. Stanczak, M. Wiczanowski, and H. Boche, *Resource Allocation in Wireless Networks: Theory and Algorithms*, NY, USA: Springer, 2006.

- [72] V. Taranalli, H. Uchikawa and P. H. Siegel, “Error Analysis and Inter-Cell Interference Mitigation in Multi-Level Cell Flash Memories,” in *Proc. IEEE Int. Conf. Commun. (ICC)*, London, Jun. 2015 , pp. 271-276.
- [73] R. Timo, K. Blackmore, and L. Hanlen, “Word-Valued Sources: An Ergodic Theorem, an AEP, and the Conservation of Entropy”, *IEEE Trans. Inf. Theory*, vol. 56, no. 7, pp. 3139-3148, Jul. 2010.
- [74] T. J. Tjalkens and F. M. J. Willems, “Variable to fixed-length codes for Markov sources,” *IEEE Trans. Inf. Theory*, vol. IT-33, pp. 246-257, 1987.
- [75] B. P. Tunstall, “Synthesis of noiseless compression codes,” Ph.D. thesis, Georgia Inst. Technol., Atlanta, GA, 1968.
- [76] O. Uchida, “Maximum generating rate of the variable-length nonuniform random number,” in *Proc. IEEE Inf. Theory Workshop*, Cairns, Australia, Sep. 2-7, 2001, pp. 141–143.
- [77] G. Ungerboeck, “Huffman shaping,” in *Codes, Graphs, and Systems*, R. Blahut and R. Koetter, Eds. Springer, 2002, ch. 17, pp. 299–313.
- [78] B. Varn, “Optimal variable length codes (arbitrary symbol cost and equal code word probability),” *Inform. Contr.*, vol. 19, pp. 289–301, 1971.
- [79] K. Visweswariah, S. R. Kulkarni and S. Verdu, “Source codes as random number generators,” *IEEE Trans. Inf. Theory*, vol. 44, no. 2, pp. 462–471, Mar. 1998.
- [80] A. Wald, *Sequential Analysis*. Courier Corporation, 1973.