UNIVERSITY OF CALIFORNIA

Los Angeles

Graph-Based Error Correcting Codes
for Modern Dense Storage Devices

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Electrical and Computer Engineering

by

Ahmed Hassan Mahmoud E Hareedy

2018

ABSTRACT OF THE DISSERTATION

Graph-Based Error Correcting Codes
for Modern Dense Storage Devices

by

Ahmed Hassan Mahmoud E Hareedy
Doctor of Philosophy in Electrical and Computer Engineering
University of California, Los Angeles, 2018
Professor Lara Dolecek, Chair

In order to meet the demands of data-hungry applications, modern data storage systems are expected to be increasingly denser. This is a challenging endeavor, and storage engineers are continuously trying to provide novel technologies. However, these new technologies are typically associated with an increase in the number and types of errors, making the goal of securing highly-reliable dense storage devices a tricky challenge.

This dissertation focuses on analyzing the errors in addition to providing novel and efficient error correcting coding schemes that are capable of overcoming the aforementioned challenge. In particular, through informed exploitation of the underlying channel characteristics of the storage device being studied, we provide frameworks for systematically generating error correcting codes with mathematical guarantees that offer performance improvements in orders of magnitude relative to the prior state-of-the-art.

First, we present a technique to predict the performance of codes given the existence of certain error-prone structures in the graph representation of these codes. Next, we introduce a general framework for the code optimization of non-binary graph-based codes, which works for various interesting channels. Finally, we derive an approach to design high performance spatially-coupled codes particularly for magnetic recording applications.

Our frameworks are based on mathematical tools drawn from coding theory and information theory, and rely on advanced mathematical techniques from probability theory, linear algebra, graph theory, combinatorics, and optimization. The proposed frameworks have a vast variety of applications that include both magnetic recording and Flash memory systems. Our frameworks lead to a practical, effective tool for storage engineers to use multi-dimensional storage devices with confidence.

The dissertation of Ahmed Hassan Mahmoud E Hareedy is approved.

Danijela Cabric

Lieven Vandenberghe

Guy Van den Broeck

Lara Dolecek, Committee Chair

University of California, Los Angeles

2018

To my parents and my brothers.

Table of Contents

Acknowledgements

When I left Egypt in 2014 to come to USA and start my Ph.D. journey at UCLA, I was an industry veteran. Thus, it has not been an easy journey by any means, and I cannot picture reaching this point without the priceless help and support of my family, my advisor, my mentors, my collaborators, and my friends at UCLA and at my motherland. Before talking about the people I am going to acknowledge, I must thank Allah for giving me the power and guidance throughout my life, and particularly in my entire Ph.D. time, to overcome the challenges and obstacles. I am forever grateful.

First, I would like to sincerely thank my advisor, Professor Lara Dolecek. Professor Dolecek has given me the opportunity to come to UCLA and work in her prestigious LORIS lab. Using her massive technical knowledge and awareness of the state-of-the-art research topics in the field, Professor Dolecek has guided me to work on impactful research problems. Professor Dolecek also has amazing decision making skills; she has constantly provided me with insightful information about when to work on a specific problem and what are the best venues to submit our results to. Having said that, she has always given me the freedom to conduct my research, which is something I highly appreciate. Moreover, Professor Dolecek has used her network of connections to introduce me to many big names in the field and to help me with my internships along with my future steps. She has also been kind and understanding throughout this journey. I have learned a lot of valuable things from Professor Dolecek that will affect my academic career in a positive way.

Next, I would like to thank my other committee members, Professor Danijela Cabric, Professor Lieven Vandenberghe, and Professor Guy Van den Broeck, for their support and for their thoughtful feedback. Special thanks to Professor Cabric for generously providing her precious help in different occasions.

Outside of UCLA, I have been lucky to know Professor Bane Vasic (University of Arizona), with whom I have had a lot of fruitful conversations. I would like to thank Professor

Vasic for his valuable advice and for his support regarding my career. Moreover, I have been honored to recently know Professor Robert Calderbank (Duke University). I have enjoyed all my technical and non-technical discussions with him, and I am already looking forward to starting my work with him soon. During my two internships, I was managed by Professor Ravi Motwani (Intel Corporation), who is an expert in coding theory. It has been my pleasure working with him, and I have collected useful knowledge to my research from these internships. Furthermore, I have had the opportunity to work with Dr. Rick Galbraith (Western Digital Corporation) on multiple papers. During this work, Dr. Galbraith provided important contributions and comments.

It has been an exceptional experience for me performing research with LORIS members Homa Esfahanizadeh, Chinmayi Lanka, Ruiyi Wu, Dr. Behzad Amiri, Dr. Clayton Schoeny, Nian Guo, and Andrew Tan. Two names from this list stand out; Homa and Chinmayi. My work with Homa will leave unforgettable memories for both of us. These memories include several publications between journal papers, conference papers, and workshop abstracts, in addition to a Memorable Paper Award. As for Chinmayi, who has kept working with me for two years after leaving UCLA, it is enough to mention that my fruitful collaboration with her has resulted in two journal papers published in prestigious venues. I would like to thank all LORIS members whom I have worked with. Moreover, the tools developed by Dr. Yuta Toriyama have been valuable in the research.

I would also like to thank other LORIS members whom I have not had the chance to do research with. They are Dr. Frederic Sala, Amirhossein Reisizadeh, Kayvon Mazooji, Shahroze Kabir, Siyi Yang, Zehui Chen, Lev Tauz, in addition to the visitors, Professor Weigang Thu and Professor Laura Conde. It has been great having all of them as colleagues. Our lab has been different from any typical engineering lab as we have spent a lot of fun time together. The wonderful outings, including going to basketball games together, will always remain in my memory. I have particularly spent a lot of time discussing various topics with Frederic, Clayton, Homa, Ruiyi, Siyi, and also Yuta.

Additionally, there are other persons I have to mention here. I appreciate all the great time I have spent with my UCLA friends and roommates, Mohammed Karmoose and Ahmed Alaa. I thank all my friends who have helped me with the logistics in LA and SF Bay Area, making my life there a lot easier. Our colleagues in the ECE Graduate Office, particularly Deeona Columbia and Ryo Arreola, have been doing a great job helping graduate students in the department. I would also like to thank my mentors at Cairo University, Professor Serag Habib and Professor Mohamed Khairy, in addition to my long-time manager at Mentor Graphics Corporation, Mohamed Selim, for all what I have learned from them and for their support. Moreover, there are my close friends in Egypt, Ramy Hosny, Amin Maher, Islam Shaboon, Rami Aliedine, Mahmoud Saber, and Reem Khairy; they have made my transition a lot smoother with their continuous and priceless help.

Finally, and most importantly, I would like to say a hearty thank you to my family; my father, Hassan Hareedy, my mother, Nadia Farghaly, and my brothers, Mohammed Hareedy and Kareem Hareedy. Even though he died long ago, my father has established for our family what everything starts from. As for my mother, she has been my first teacher and the most important person throughout my life. I am blessed with her tremendous love, her wise advice, and her sincere prayers. I am simply blessed with the existence of my mother in my life, and I thank her for everything. My brothers have always been my ultimate friends. I hope that I have done something they all will be proud of.

My research was sponsored by grants from the Advanced Storage Technology Consortium - the International Disk Drive Equipment and Material Association (ASTC-IDEMA) along with grants from the National Science Foundation (NSF), which are CCF 1029030, CCF-CAREER 1150212, and CCF-BSF 1718369. In my last year, my research was also supported by the UCLA Dissertation Year Fellowship.

Vita

2006	Bachelor in Electronics and Communications Engineering Cairo University
2006-2014	Senior Development Engineer Deep Sub-Micron Division, Mentor Graphics Corporation
2011	Master of Science in Electronics and Communications Engineering Cairo University
2014-2018	Research Assistant Electrical and Computer Engineering Department University of California, Los Angeles
2015, 2017	Error Control Coding Architect Non-Volatile Memory Solutions Group, Intel Corporation
2015	Best Paper Award IEEE Global Communications Conference
2017-2018	Dissertation Year Fellow University of California, Los Angeles
2017	Henry Samueli Excellence in Teaching Award Electrical Engineering Department, University of California, Los Angeles
2018	Memorable Paper Award Non-Volatile Memories Workshop

Selected Publications

A. Hareedy, B. Amiri, R. Galbraith, S. Zhao, and L. Dolecek, "Non-binary LDPC code optimization for partial-response channels," in *Proc. IEEE Global Commun. Conf.*

(GLOBECOM), San Diego, CA, USA, Dec. 2015, pp. 1–6.

A. Hareedy, B. Amiri, R. Galbraith, and L. Dolecek, "Non-binary LDPC codes for magnetic recording channels: error floor analysis and optimized code design," *IEEE Trans. Commun.*, vol. 64, no. 8, pp. 3194–3207, Aug. 2016.

A. Hareedy, C. Lanka, and L. Dolecek, "A general non-binary LDPC code optimization framework suitable for dense Flash memory and magnetic storage," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 9, pp. 2402–2415, Sep. 2016.

A. Hareedy, H. Esfahanizadeh, and L. Dolecek, "High performance non-binary spatially-coupled codes for flash memories," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Kaohsiung, Taiwan, Nov. 2017, pp. 229–233.

H. Esfahanizadeh, **A. Hareedy**, and L. Dolecek, "Finite-length construction of high performance spatially-coupled codes via optimized partitioning and lifting," accepted at *IEEE Trans. Commun.*, doi: 10.1109/TCOMM.2018.2867493, Aug. 2018.

A. Hareedy, C. Lanka, N. Guo, and L. Dolecek, "A combinatorial methodology for optimizing non-binary graph-based codes: theoretical analysis and applications in data storage," accepted at *IEEE Trans. Inf. Theory*, doi: 10.1109/TIT.2018.2870437, Sep. 2018.

H. Esfahanizadeh, **A. Hareedy**, and L. Dolecek, "Multi-dimensional spatially-coupled code design through informed relocation of circulants," accepted at *56th Annual Allerton Conf. Commun., Control, and Computing*, Monticello, IL, USA, Oct. 2018. [Online]. Available: http://arxiv.org/abs/1809.04798

A. Hareedy, H. Esfahanizadeh, A. Tan, and L. Dolecek, "Spatially-coupled code design for partial-response channels: optimal object-minimization approach," accepted at *IEEE Global Commun. Conf. (GLOBECOM)*, Abu Dhabi, UAE, Dec. 2018. [Online]. Available: http://arxiv.org/abs/1804.05504

CHAPTER 1

Introduction

We are living in the age of big data, which requires modern storage devices to be dense. For example, two dimensional magnetic recording (TDMR) devices can be used to store up to 1 terabytes per square inch. These dense storage devices operate at extremely low frame error rates (FERs) that can only be achieved through novel, efficient error correcting coding (ECC) techniques. Among those efficient ECC techniques, arise graph-based ECC techniques, which are the topic of this dissertation.

Low-density parity-check (LDPC) codes are graph-based parity-check codes that are described by a sparse parity-check matrix $\mathbf{H}$ [1]. They were first introduced by Gallager in 1962. LDPC codes are known to have capacity approaching performance. Moreover, their simple iterative decoding, which is based on message passing algorithms, makes them even more attractive for being used in different applications.

Due to their excellent performance in other applications, LDPC codes are now actively being considered for modern dense storage devices, such as hard disk drives and multi-level Flash memories. Moreover, it is well known that non-binary LDPC (NB-LDPC) codes outperform their binary counterparts [2]. The major flaw of LDPC codes (binary and non-binary) is the error floor phenomenon caused by absorbing sets (ASs) [3], which are detrimental subgraphs in the graph of the code. The error floor phenomenon is characterized by a change in the slope of the frame error rate (FER) curve (to be worse) in the region where the channel is "good" (e.g., high signal to noise ratio (SNR) or low raw bit error rate

(RBER) regions). While the error floor of binary LDPC codes has been well studied in the literature, e.g., [3, 4, 5, 6], this problem for NB-LDPC codes is far less explored. First results in the error floor of NB-LDPC codes include [7], [8], [9], and [10].

The first research problem we discuss in this dissertation is about error floor analysis of NB-LDPC codes over magnetic recording channels. Practical 1-D magnetic recording (MR) channels are commonly modeled as partial-response (PR) channels [11, 12]. Since MR applications must operate at very low bit error rates, LDPC codes have already been studied in the context of PR channels; [13], [14], and [15] have investigated binary LDPC codes and their error floors in this context. On the contrary, and despite their potential, not much work has been done on NB-LDPC codes for PR channels. First results on this topic include [16] and [17]. The work in [16] investigated the progressive-edge-growth (PEG) algorithm to design NB-LDPC codes for PR channels. The authors in [17] studied non-binary quasi-cyclic LDPC (NB-QC-LDPC) codes for PR channels.

The time-consuming nature of error floor simulations has triggered research in error floor prediction. For binary LDPC codes used for additive white Gaussian noise (AWGN) channels, the authors in [18] presented a general method to estimate the error floor based on the dominant error objects using biased simulations (importance sampling). While many works investigated the error floor prediction for binary LDPC codes over AWGN channels, including [5], [19], [20], and [21], very few explored this problem in the case of PR channels; notable examples are [14], [22], and [23]. To the best of our knowledge, no previous literature work has studied error floor prediction for NB-LDPC codes over PR channels.

The second research problem we address in this dissertation is about deriving a general framework that can optimize NB-LDPC codes for a wide span of applications including Flash memories incorporating highly asymmetric channels. The authors in [10] studied a subclass of non-binary ASs (NB ASs), that is the NB elementary ASs (EASs), and showed that EASs are the principal cause of the error floor of NB-LDPC codes over additive white Gaussian noise (AWGN) channels. We demonstrate that the nature of the detrimental objects which dominate the error floor region of NB-LDPC codes critically depends on the channel of interest. As a result, we conclude that using NB-LDPC codes that are optimized for AWGN channels is not appropriate for practical Flash channels (resp., PR channels) because of the asymmetry (resp., the intrinsic memory) the system incorporates. While the operational asymmetry in Flash memory systems is well documented [24, 25], the common code-design approach is still to directly apply LDPC codes that are optimized for AWGN-like channels [26, 27, 28], which does not give the best performance.

Next, we provide the in-depth theoretical analysis of the general NB-LDPC code optimization framework mentioned in the above paragraph. In particular, the detrimental objects that dominate the error floor region of NB-LDPC codes over asymmetric channels are characterized, and the optimality of the aforementioned framework, in terms of complexity, is proved. This optimization framework removes a detrimental object via careful processing of its edge weights. Here, we also address questions related to what is the minimum number of edge weights to be changed and how to select the edges. Extensions to the framework are also proposed. Such extensions include how to achieve additional performance gains for NB-LDPC codes with even column weights (variable node degrees) and include optimizing non-binary spatially-coupled codes for asymmetric channels.

The last research problem we tackle is about the design of high performance binary and non-binary spatially-coupled (SC) codes. Spatially-coupled (SC) codes [29, 30, 31] are graph-based codes constructed by partitioning an underlying block code into components of the same size, then rewiring these components multiple times [32]. In this work, the underlying block codes, and consequently our constructed SC codes, are circulant-based (CB) codes. SC codes offer not only complexity/latency gains (if windowed decoding [33] is used), but also an additional degree of freedom in the code design; this added flexibility is achieved via partitioning of the parity check matrix of the underlying block code. This observation makes SC codes receive an increasing level of attention in multiple applications. Contiguous [32] and non-contiguous [34, 35, 36] partitioning schemes were introduced in the literature for various applications.

Recently, we have introduced the optimal overlap (OO), circulant power optimizer (CPO) approach to design SC codes with superior performance for AWGN [37] and practical asymmetric Flash [38] channels. The OO partitioning operates on the protograph to compute the optimal set of overlap parameters that characterizes the partitioning. The CPO operates on the unlabeled graph (edge weights are set to 1's) to adjust the circulant powers. The objective is to minimize the number of instances of a common substructure that exists in several detrimental objects. If the SC code is binary, the unlabeled graph is the final graph. If the SC code is non-binary, the edge weights are optimized after applying the OO-CPO approach. Here, we aim at extending the OO-CPO approach to construct high performance SC codes for magnetic recording applications.

1.1 Outline of Contributions

In this section, we provide a summary of the contributions that will be detailed in each of the following five chapters of the dissertation. Our related publications in the references are [39] and [40] for Chapter 2, [41] and [42] for Chapter 3, [43] for Chapter 4, and [44] and [45] for Chapter 5, in addition to the relevant publications [32], [34], [35], [37], [38], [46], [47], and [48]. Further details about the described work in this dissertation can be found in these publications.

1.1.1 Chapter 2 Contributions

In this chapter, we offer a detailed study of the error floor performance of structured NB-LDPC codes over PR channels. In particular, we show that the error profile in the error floor region over PR channels is qualitatively different from the error profile over AWGN channels. A subset of absorbing sets (ASs) [3], called balanced absorbing sets (BASs), is introduced and is shown to be the dominant error type over PR channels. Based on these

BASs, we propose a method to accurately predict the error floor of NB-LDPC codes over PR channels. To accurately perform such prediction, we add two major refinements to the prediction method introduced in [23]: incorporation of the inter-symbol interference (ISI) (effect of channel density) in the method and accurate modeling of the pattern-dependent jitter. Simulation results show that our method is capable of estimating the error floor of NB-LDPC codes over PR channels within only 0.2 of an order of magnitude of the actual Monte Carlo (MC) simulations, and can be used to predict low error floor levels (e.g., $< 10^{-12}$). Furthermore, we provide the theoretical analysis of how to efficiently remove a BAS from the graph of an NB-LDPC code.

1.1.2 Chapter 3 Contributions

In this chapter, we re-visit the existing definitions of combinatorial objects, such as ASs and elementary absorbing sets (EASs) [3, 10], that were proved to be useful in the error floor analysis of NB-LDPC codes over AWGN channels. By recognizing that the existing definitions are insufficient to describe the errors for asymmetric channels, we introduce a more finely specified combinatorial object: the general absorbing sets of type two (GASTs). Our NB-LDPC code optimization objective for aggressively asymmetric channels then becomes the removal of GASTs. Through a succinct matrix-theoretic representation, we express a GAST as a set of submatrices, which we call weight consistency matrices (WCMs). By forcing the null spaces of the resultant WCMs to have a particular property, we provably remove detrimental GASTs from the graph representation of the code. We also demonstrate that the WCM definition can be customized to accurately capture the properties of other subclasses of GASTs, e.g., EASs and balanced absorbing sets of type two (BASTs).

This work offers the first theoretical framework, which we call the WCM framework, for the analysis and design of NB-LDPC codes over realistic storage channels with asymmetry, e.g., the normal-Laplace mixture (NLM) Flash channel [24]. We show the effectiveness of the WCM framework over many channels with different characteristics. We present results for the NLM channel, the Cai Haratsch Mutlu Mai (CHMM) Flash channel [25], the PR channel [11, 12], and the AWGN channel. Over all these channels, the codes optimized using the WCM framework outperform unoptimized codes by a minimum of 1, and up to nearly 2 orders of magnitude in the uncorrectable bit error rate (UBER) [49] or the frame error rate (FER). Furthermore, over asymmetric channels, the codes optimized using the WCM framework also outperform the codes optimized for symmetric channels.

1.1.3 Chapter 4 Contributions

In this chapter, we define the unlabeled GAST tree to describe the underlying topology of a GAST, where the leaves of this tree represent the WCMs of the GAST. Using this tree, we prove the optimality of the WCM framework by demonstrating that the framework indeed operates on the minimum possible number of matrices to remove the detrimental object. We also compute the exact number of WCMs associated with a GAST in different cases. We further make a comparison with a suboptimal idea.

Next, we propose a comprehensive analysis of the removal process of GASTs. We start off with discussing the dimensions of the null spaces of WCMs; these null spaces play the central role in the identification and removal of a GAST. Then, we derive the best that can be done to process a short WCM during the GAST removal process. Finally, we provide the minimum number of edge weight changes needed to remove a GAST, along with how to appropriately select the edges and the new weights.

Moreover, we introduce new combinatorial objects that capture the majority of the non-GAST detrimental objects in the error floor region of NB-LDPC codes that have even column weights over asymmetric Flash channels. We define oscillating sets of type two (OSTs). Furthermore, we expand the analysis of GASTs in Chapter 3 to cover OSTs, describing how the WCM framework can be customized to remove OSTs, after GASTs have been removed, to achieve additional performance gains.

We also extend the scope of the WCM framework by using it to optimize codes with different properties and for various applications. Specifically, we demonstrate the benefits of the WCM framework in optimizing column weight 5 codes, codes used over Flash channels with additional soft information, and spatially-coupled codes. The performance gains achieved via the WCM framework range between 1 and nearly 2.5 orders of magnitude in the error floor region over interesting channels.

1.1.4 Chapter 5 Contributions

In this chapter, we propose an approach to construct high performance SC codes for PR channels. Unlike the case of AWGN and Flash channels (see [37] and [38]), the common substructure, whose number of instances we seek to minimize, in the case of PR channels can appear in different ways in the protograph of the SC code, making the optimization problem of partitioning considerably more challenging. For that reason, we introduce the concept of the pattern, which is a configuration in the protograph that can result in instances of the common substructure in the unlabeled graph of the SC code after lifting. We derive an optimization problem, in which we express the weighted sum of the counts (numbers of instances) of all patterns in terms of the overlap parameters, which are the parameters that characterize the partitioning. Then, we compute the optimal set of overlap parameters (OO) that minimizes this sum. Moreover, we propose the necessary modifications to the CPO algorithm presented in [37] and [38] to make it suitable for the common substructure in the case of PR channels. We demonstrate the gains achieved by our OO-CPO (-WCM for NB SC codes) approach through tables and performance plots that compare our codes not only with SC codes, but also with CB block codes of the same length and rate.

CHAPTER 2

A Technique for Error Floor Prediction

2.1 Introduction

Practical 1-D magnetic recording (MR) channels are commonly modeled as partial-response (PR) channels [11, 12]. Since MR applications must operate at very low bit error rates, low-density parity-check (LDPC) codes have already been studied in the context of PR channels; [13], [14], and [15] have investigated binary LDPC codes and their error floors in this context. While it has long been known that non-binary LDPC (NB-LDPC) codes outperform their binary counterparts on additive white Gaussian noise (AWGN) channels [2], not much work has been done on NB-LDPC codes for PR channels. First results on this topic include [16] and [17]. The work in [16] investigated the progressive-edge-growth (PEG) algorithm to design NB-LDPC codes for PR channels. The authors in [17] studied non-binary quasi-cyclic LDPC (NB-QC-LDPC) codes for PR channels. These efforts demonstrated the potential of using NB-LDPC codes for PR channels.

The time-consuming nature of error floor simulations has triggered research in error floor prediction. For binary LDPC codes used for AWGN channels, the authors in [18] presented a general method to estimate the error floor based on the dominant error objects and using biased simulations (importance sampling). While many works investigated the error floor prediction for binary LDPC codes over AWGN channels, including [5], [19], [20], and [21], very few explored this problem in the case of PR channels; notable examples are [14], [22], and [23]. To the best of our knowledge, no previous literature work has studied error floor prediction for NB-LDPC codes over PR channels.

Our contributions in this work are:

1. We offer a detailed study of the error floor performance of structured and regular NB-LDPC codes over PR channels. In particular, we show that the error profile[1] in the error floor region over PR channels is qualitatively different from the error profile over AWGN channels. A subset of absorbing sets (ASs) [3], called balanced absorbing sets (BASs), is introduced and is shown to be the dominant error type over PR channels. Certain qualitative and quantitative properties of BASs are then presented.

2. Based on these BASs, we propose a method to accurately predict the error floor of NB-LDPC codes over PR channels. To accurately perform such prediction, we add two major refinements to the prediction method introduced in [23]: incorporation of the inter-symbol interference (ISI) (effect of channel density) in the prediction method and accurate modeling of the pattern-dependent jitter. Simulation results show that our method is capable of estimating the error floor of NB-LDPC codes over PR channels within only 0.2 of an order of magnitude of the actual Monte Carlo (MC) simulations, and can be used to predict low error floor levels (e.g., $< 10^{-12}$).

3. We provide the theoretical analysis of how to efficiently remove a BAS from the graph of an NB-LDPC code. In particular, we discuss the minimum number of edge weight changes needed to remove a BAS. The NB-LDPC code optimization algorithm and simulation results showing the performance gains that result from removing BASs from the graphs of NB-LDPC codes will be presented in Chapter 3 and Chapter 4. More details can be found in [40] and [47].

In Section 2.2, we describe the detrimental structures contributing to the error floor of NB-LDPC codes over PR channels, and present a theoretical discussion on such structures.

[1]For a given code, an error profile refers to the distribution (numbers of occurances) of decoding errors associated with certain configurations at a specified simulation setup.

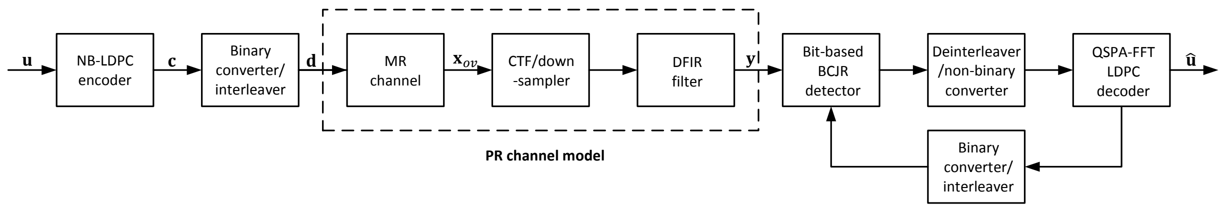


Figure 2.1: System model for 1-D MR channel utilizing an NB-LDPC code.

Our error floor prediction method is described in Section 2.3 along with simulation results showing its effectiveness. In Section 2.4, we present the techniques used to remove BASs. The chapter ends with concluding remarks in Section 2.5.

2.2 Error Profile of NB-LDPC Codes over PR Channels

In this section, after introducing the system model, we discuss the error floor of NB-LDPC codes over PR channels by contrasting it with the error floor over AWGN channels. Additionally, we show the effect of iterations between the detector and the decoder on the error floor performance, and, most importantly, introduce new detrimental objects: balanced absorbing sets (BASs).

2.2.1 System Model

We first describe the system model for the NB-LDPC coded PR channel which will be used throughout this chapter. Vectors in this chapter are row vectors. Consider an NB-LDPC code over GF(q), $q = 2^p$, with block length m and dimension k. The system model, shown in Fig. 2.1, has the following components:

Encoding: The information sequence $\mathbf{u} = [u_1\ u_2\ \ldots\ u_k] \in \mathrm{GF}(q)^k$ is encoded into a codeword $\mathbf{c} = [c_1\ c_2\ \ldots\ c_m] \in \mathrm{GF}(q)^m$. In this chapter, we focus on regular NB-LDPC codes with implementation-friendly structures, e.g., non-binary protograph-based LDPC (NB-PB-LDPC) codes (also circulant-based) presented in [8] and also in [9] and [17].

Transmission: The codeword $\mathbf{c}$ is converted to a binary sequence that is modulated and interleaved pseudo-randomly into the sequence of data $\mathbf{d} = [d_1\ d_2\ \ldots\ d_n]$ (e.g., $d_j \in \{-1, +1\}$ for binary phase shift keying (BPSK)) with $n = pm$. The vector $\mathbf{d}$ is then written onto the MR channel.

Channel: The 1-D MR channel includes ISI resulting from the read-head sensitivity (see [50] and [51]), as well as jitter and electronic noise. The normalized density [50] of the MR channel is commonly set to 1.4. We consider other values of the MR channel density as well. The 1-D MR channel output is the oversampled sequence $\mathbf{x}_{ov}$.

Filtering: Continuous-time filtering (CTF) and down-sampling are applied sequentially to the sequence $\mathbf{x}_{ov}$. The resulting sequence is passed through a digital finite impulse response (DFIR) filter. CTF and DFIR units are used to achieve the PR equalization target. According to industry recommendations, the default target we use is [8 14 2]. As we will show later, we also examined several other PR targets; for all these targets, our simulation results were consistent.

The MR channel, CTF, and DFIR units are considered altogether as the PR channel.

Detection/Decoding: A finite-precision fast Fourier transform based q-ary sum-product algorithm (FFT-QSPA) LDPC decoder [52] and a Bahl Cocke Jelinek Raviv (BCJR) detector [53], that incorporates pattern-dependent noise prediction (PDNP) [54], are used to iteratively estimate the sequence $\mathbf{u}$.

We prefer to use a bit-based detector [16] because of the high complexity of symbol-based detectors. Thus, in order to properly address the correlation between bits due to the ISI, we adopt an appropriately designed bit-based interleaver to distribute correlated bits to different symbols [16] such that the correlation between bits within the same symbol is nearly removed.

The iterations executed inside the LDPC decoder are referred to as **local iterations**. Each outer looping between the detector and the decoder is referred to as one **global iteration**. In between two consecutive global iterations, the decoder executes its prescribed number of local iterations (or fewer, if a codeword is reached).

2.2.2 Background and Motivating Examples

NB-LDPC codes are defined over bipartite graphs called **Tanner graphs**. A non-zero value $\in$ GF(q) in the parity-check matrix of an NB-LDPC code is mapped into an **edge** connecting a variable node (VN) to a check node (CN) in the Tanner graph of that code. This non-zero value $\in$ GF(q) is called the **edge weight**. The **unlabeled graph** is generated by replacing all edge weights by ones. Based on this brief introduction, we first recall the following definitions:

Definition 1. *(cf. [3] and [10]) Consider a subgraph induced by a subset $\mathcal{V}$ of VNs in the Tanner graph of an NB-LDPC code. Set all the VNs in $\mathcal{V}$ to values $\in$ GF(q)$\backslash\{0\}$ and set all other VNs to 0. The set $\mathcal{V}$ is said to be an (a, b)* ***absorbing set (AS)*** *if the size of $\mathcal{V}$ is a, the number of neighboring unsatisfied CNs is b, and each VN in $\mathcal{V}$ is connected to strictly more satisfied than unsatisfied neighboring CNs, for some set of VN values.*

Definition 2. *(cf. [10] and [55]) An* ***elementary absorbing set*** *is an absorbing set with each of its neighboring satisfied CNs having two edges connected to the set, and each of its neighboring unsatisfied CNs having exactly one edge connected to the set.*

Remark 1. *(cf. [10]) For a* ***non-binary elementary absorbing set****, the unlabeled subgraph is a binary AS, and all the cycles of this subgraph satisfy the following weight equation:*

$$\prod_{i=1}^{p} \rho_{2i-1} = \prod_{i=1}^{p} \rho_{2i} \text{ over GF}(q), \tag{2.1}$$

where ρ_j's, $1 \leqslant j \leqslant 2p$, are edge weights and $2p$ is the cycle length. Note that for a non-binary AS which is ***non-elementary****, Equation (2.1) does not necessarily hold for all of its cycles.*

The following examples illustrate that the error profiles for PR and AWGN channels are qualitatively different; we observed similar behavior for other channel and code parameters. We define e_{wt} as the error weight.

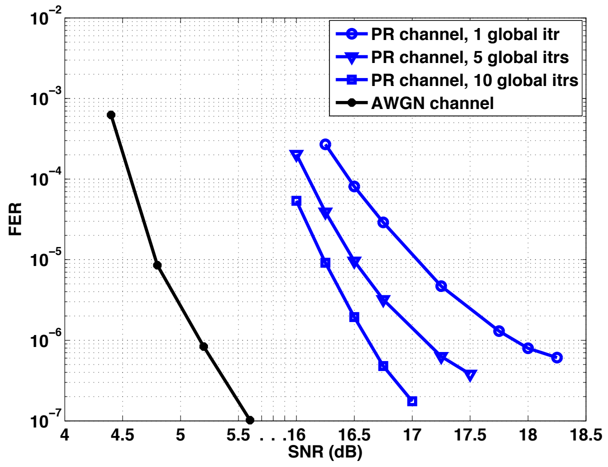


Figure 2.2: FER curves of Code 2.1 over PR (with the default target) and AWGN channels.

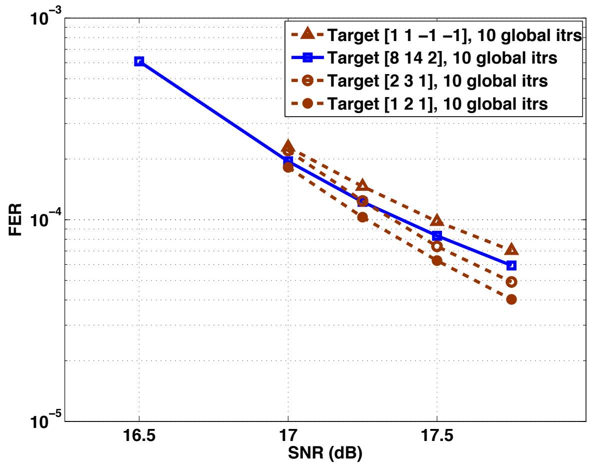


Figure 2.3: FER curves of Code 2.2 over PR channels with different equalization targets.

Example 1. *Consider Code 2.1, which is an NB-PB-LDPC code (designed according to [8] and [9]) with block length* $n = 4{,}092$ *bits, rate* $R \approx 0.87$*,* $q = 16$*, and column weight* $\gamma = 4$*. Fig. 2.2 shows the performance curves of Code 2.1 over PR and AWGN channels. Tables 2.1 and 2.2 show the error profiles at representative SNR values. The tables show that the dominant error in case of the PR channel is the* $(6, 2)$ *AS while the dominant errors in case of the AWGN channel are the* $(4, 4)$*,* $(6, 4)$*, and* $(6, 6)$ *ASs. (BAS and UBAS in Table 2.1 refer to balanced and unbalanced ASs, respectively, which we will define shortly.)*

Example 2. *Consider Code 2.2, which is an NB-PB-LDPC code (designed according to [8] and [9]) with* $n = 1058$ *bits,* $R \approx 0.87$*,* $q = 4$*, and* $\gamma = 3$*. Fig. 2.3 shows the performance*

Table 2.1: Error profile of Code 2.1 over the PR channel at 10 global iterations, SNR = 16.75 dB, FER $\approx$ 4.79e$-$7.

Error Weight	Low ($e_{\text{wt}} < 4$)	Medium ($4 \leqslant e_{\text{wt}} < 12$)				High ($e_{\text{wt}} \geqslant 12$)	
Type	0	$(6,2)$ BAS	Other BAS	UBAS	Random	BAS Comb.	Random
Count		86	5	2	2	4	1

Table 2.2: Error profile of Code 2.1 over the AWGN channel, SNR = 5.60 dB, FER $\approx$ 1.05e$-$7.

Error Type	$(4,4)$	$(5,2)$	$(6,2)$	$(6,4)$	$(6,6)$	$(7,4)$	Other
Count	44	4	7	17	12	4	12

curves of Code 2.2 over PR channels with the equalization targets $[1\ 1\ -1\ -1]$*,* $[8\ 14\ 2]$ *(default),* $[2\ 3\ 1]$*, and* $[1\ 2\ 1]$*. The first and the fourth targets are known as extended PR (EPR4) and PR2 targets, respectively [22]. Our simulations show that the dominant error in case of PR channels for all the four targets is the* $(6,0)$ *AS while it is mainly the* $(3,3)$ *AS in case of the AWGN channel.*

Our extensive simulations for different code parameters and channel parameters, including the PR target as shown in Example 2, confirm that indeed the PR channel error profile is different from the AWGN channel error profile.

2.2.3 New Definitions of Detrimental Objects

Motivated by the error profile differences between AWGN and PR channels shown in the last subsection, we now introduce new combinatorial definitions aimed at capturing the decoding errors under the PR channel.

Let $g = \left\lfloor \frac{\gamma-1}{2} \right\rfloor$ for a given column weight γ. Aided by the simulation results of the type shown in Examples 1 and 2, we classify ASs as follows, depending on the relative value of b.

Definition 3. *An* (a,b) *absorbing set with* $0 \leqslant b \leqslant \left\lfloor \frac{ag}{2} \right\rfloor$ *is defined as a* ***balanced absorbing set (BAS)****, while an* (a,b) *absorbing set with* $\left\lfloor \frac{ag}{2} \right\rfloor < b \leqslant ag$ *is defined as an* ***unbalanced absorbing set (UBAS)****.*

An AS with fewer unsatisfied CNs b (i.e., $0 \leqslant b \leqslant \left\lfloor \frac{ag}{2} \right\rfloor$) has more immunity against the detector-decoder looping compared with an AS with $\left\lfloor \frac{ag}{2} \right\rfloor < b \leqslant ag$, which may be resolved with sufficient looping. This observation motivates us to call the former balanced (i.e., more stable and more difficult to correct) and the latter unbalanced (i.e., more unstable and easier to correct). BASs play a critical role in the error profile of PR channels as we shall see later. In the following sections (and chapters), we will show how BASs can be used to accurately estimate the error floor performance of NB-LDPC codes over PR channels, and to effectively optimize such codes.

Examples of BASs include $(6,0)$ AS for $\gamma = 3$ ($g = 1$), and $(8,3)$ and $(6,2)$ ASs for $\gamma = 4$ ($g = 1$). Codewords are special cases of BASs ($b = 0$). Examples of UBASs include $(3,3)$ AS for $\gamma = 3$, and $(6,4)$ and $(6,6)$ ASs for $\gamma = 4$.

2.2.4 Effect of Global Iterations

In addition to having an error profile different from the AWGN channel case, we further observe that in the PR channel case, the error profile changes as a function of the number of global iterations. For Code 2.1 of Example 1, Fig. 2.2 shows the performance curves over the PR channel at 1, 5, and 10 global iterations. This example serves to illustrate how the error profile changes as a function of the number of global iterations; we have observed a similar behavior for other codes and system parameters. For this example, we tabulate the errors in Tables 2.1, 2.3, and 2.4 for 10, 1, and 5 global iterations, respectively. Observe that, when the receiver executes a low number of global iterations (say for example, 5), the decoding errors are usually due to BASs and their high-weight combinations. In fact, through all our simulations, we observed that high-weight errors are mostly combinations of two or more low-weight BASs. This observation motivates Definition 4.

We say that a BAS is a **dominant BAS** if it causes most of the uncorrectable errors in the error floor region of the NB-LDPC code over the channel of interest. For example, the dominant BAS is the $(6,2)$ BAS for Code 2.1 over the PR channel (see Table 2.1).

Let s be the weight of the smallest dominant BAS for a given code in the error floor region over the PR channel (for a sufficient number of global iterations).

Definition 4. *For a given code, an (a, b) balanced absorbing set that has $s \leqslant a < 2s$ is defined as a **first-order balanced absorbing set (FOBAS)**.*

Intuitively, BASs that have $e_{\text{wt}} \geqslant 2s$ are mostly formed as combinations of two or more FOBASs, hence we choose the name first-order BASs for the latter.

A lower bound on s can be evaluated based on the size of of the smallest possible BAS given the structure of the code. However, care must be taken in PR channels as the smallest possible BAS need not be the dominant BAS, as our examples show.

In PR systems, it is useful to classify decoding errors into:

1. Low-weight errors: Such errors are of weight less than the minimum AS size (a_{min}). These errors occur due to the intrinsic memory of the MR channel, and are typically prevented by the CTF and DFIR units. Increasing the number of global iterations is generally sufficient to eliminate these decoding errors.

2. Medium-weight errors: These decoding errors have weight e_{wt} such that $a_{\text{min}} \leqslant e_{\text{wt}} < 2s$. As the number of global iterations increases, medium-weight errors dominate the error profile and they are overwhelmingly BASs.

3. High-weight errors: These decoding errors have weight $e_{\text{wt}} \geqslant 2s$. High-weight errors are either random errors or combinations of BASs. As the number of global iterations increases, random errors are mostly resolved as are BAS combinations that are formed from overlapping BASs. The overall effect is the reduction of high-weight errors as the number of global iterations increases. For a sufficient number of global iterations, these high-weight errors are typically combinations of FOBASs.

For a $\gamma = 4$ NB-LDPC code, it was shown in [10] that $a_{\text{min}} = 4$. As a result, Code 2.1 of Example 1 has $a_{\text{min}} = 4$ and $s = 6$. Thus, low-weight errors have $e_{\text{wt}} < 4$, medium-weight

Table 2.3: Error profile of Code 2.1 over the PR channel at 1 global iteration, SNR = 18.00 dB, FER = 7.96e−7.

Error Weight	Low ($e_{\mathrm{wt}} < 4$)	Medium ($4 \leqslant e_{\mathrm{wt}} < 12$)			High ($e_{\mathrm{wt}} \geqslant 12$)	
Type	15	BAS	UBAS	Random	BAS Comb.	Random
Count		0	13	27	13	32

Table 2.4: Error profile of Code 2.1 over the PR channel at 5 global iterations, SNR = 17.25 dB, FER = 6.30e−7.

Error Weight	Low ($e_{\mathrm{wt}} < 4$)	Medium ($4 \leqslant e_{\mathrm{wt}} < 12$)				High ($e_{\mathrm{wt}} \geqslant 12$)	
Type	0	$(6,2)$ BAS	Other BAS	UBAS	Random	BAS Comb.	Random
Count		47	5	5	11	23	9

errors have $4 \leqslant e_{\mathrm{wt}} < 12$, and high-weight errors have $e_{\mathrm{wt}} \geqslant 12$. Tables 2.1, 2.3, and 2.4 confirm the trends mentioned above for low, medium, and high-weight errors as the number of global iterations increases.

The change in the error profile as a function of the number of global iterations can be intuitively explained as follows. First, the increase in the number of global iterations enables the detector to provide sufficient innovation at the decoder input to resolve AS errors that are on the brink of instability, which we classified as UBASs. As a result, the remaining, unresolved errors are due to BASs. Second, due to the memory in the PR channel system, a wrong belief at a VN negatively impacts nodes that are adjacent to it. With a sufficient number of VNs having wrong beliefs, a BAS decoding error occurs. With the memory in the system, these errors propagate to adjacent VNs that themselves form other BASs. As a result, high-weight errors containing several small BASs (which we refer to as FOBASs) are observed. Increasing the number of global iterations results in removing BAS combinations caused by BASs overlap. Clearly, these fine-grained decoding error types do not exist in memoryless, AWGN-based systems.

Remark 2. *The increase in the number of global iterations can resolve the majority of high-weight errors (see e.g., Example 1, Table 2.1); however, solely relying on the increase in the* *number of global iterations is generally not a preferred strategy in practice because of the added decoding latency. In our code design approach, we will therefore focus on removing FOBASs (and consequently their combinations).*

2.2.5 Preparing the List of Problematic Objects

In this subsection, we establish some properties of BASs and FOBASs. In particular, we count all the possible pairs that can result in BASs and those that can result in FOBASs. We will refer to these counts later in the error floor prediction method.

Recall that $g = \left\lfloor \frac{\gamma-1}{2} \right\rfloor$ for a given column weight γ, and s is the weight of the smallest dominant BAS for a given code in the error floor region over the PR channel.

Lemma 1. *For a given value of a, the number of (a, b) pairs that can result in a BAS is:*

$$M_{\text{BAS}}|_a = \left\lfloor \frac{ag}{2} \right\rfloor + 1. \tag{2.2}$$

For a given value of s, the number of (a, b) pairs that can result in an FOBAS can be approximated as:

$$M_{\text{FOBAS}}|_s \approx \frac{3s^2 g}{4}. \tag{2.3}$$

Proof. The proof of (2.2) follows from Definition 3 of a BAS:

$$M_{\text{BAS}}|_a = \sum_{b=0}^{\left\lfloor \frac{ag}{2} \right\rfloor} 1 = \left\lfloor \frac{ag}{2} \right\rfloor + 1.$$

To obtain (2.3), we first consider the case when both g and s are even. Recall the definition of an FOBAS (Definition 4). Combining that definition ($s \leqslant a \leqslant 2s - 1$) with (2.2) gives:

$$\begin{aligned}
M_{\text{FOBAS}}|_s &= \sum_{a=s}^{2s-1}\left(\frac{ag}{2}+1\right) = \left(\frac{sg}{2}+1\right)+\left(\frac{(s+1)\,g}{2}+1\right)\\
&+\left(\frac{(s+2)\,g}{2}+1\right)+\cdots+\left(\frac{(2s-1)\,g}{2}+1\right)\\
&=\frac{g}{2}\left[s+(s+1)+(s+2)+\cdots+(2s-1)\right]+s\\
&=\frac{g}{2}\left[\frac{s}{2}\left(s+2s-1\right)\right]+s=\frac{g}{2}\left[\frac{3s^2}{2}-\frac{s}{2}\right]+s\\
&=\frac{3s^2g}{4}+\left(1-\frac{1}{4}g\right)s\approx\frac{3s^2g}{4}.
\end{aligned}$$

The same procedure can be followed for other choices of the values of g and s. For example, if $g=1$ and s is even, $M_{\text{FOBAS}}|_s=\frac{3s^2}{4}+\frac{s}{2}$ (exact equation). It can be shown that the dominant term, $\frac{3s^2g}{4}$, remains the same in all cases. □

Example 3. *Consider the case where $s=6$ and $g=\left\lfloor\frac{\gamma-1}{2}\right\rfloor=1$ (as in Example 1). In this case, the total number of $(6,b)$ pairs*[2]

Remark 3. *Lemma 1 provides the count of the (a,b) candidate pairs which are problematic for PR channels. If there are multiple non-isomorphic configurations with the same values of (a,b), they should all be accounted for during the code optimization process. Nonetheless, Lemma 1 provides a first-order characterization of BASs and FOBASs.*

Remark 4. *Ensemble-wide count of the number of (a,b) binary elementary ASs is derived in [55] and, for a given (a,b) binary elementary AS, the fraction of edge weight assignments that can result in a non-binary elementary AS is given in [10].*

2.3 Error Floor Prediction Method

In this section, we describe our error floor prediction method. Our method is inspired by the approach in [23], which is used for binary LDPC codes over EPR4 channel, with the

[2]These pairs are $(6,0)$, $(6,1)$, $(6,2)$, and $(6,3)$. that can result in a BAS is $M_{\text{BAS}}|_{a=6}=4$. The exact total number of (a,b) pairs[3] that can result in an FOBAS is $M_{\text{FOBAS}}|_{s=6}=\frac{3(6^2)}{4}+\frac{6}{2}=30$.

following refinements that capture the effects of the MR channel:

1. Accounting for the ISI resulting from the MR read-head sensitivity.
2. More accurate modeling of the pattern-dependent jitter.

Additionally, we extend the method in [23] from binary LDPC to NB-LDPC codes, generalize it from EPR4 target to any PR target, and change the error objects of interest from the broadly defined trapping sets to the more refined FOBASs.

2.3.1 A Theoretical Description of the Proposed Method

The objective of error floor prediction methods that are based on importance sampling is to reduce the number of required MC simulations by proposing a representative weighting factor that weighs the outcome of a decoding error. Our goal is to accurately estimate the error floor performance via presenting a weighting factor that captures the MR channel ISI and jitter. To reach this goal, we perform the following steps:

1. We approximate the FER using a union bound of the dominant decoding error events.
2. We model the main effects of the MR channel (ISI and jitter) and derive its output.
3. Aided by the MR channel effects modeled in Step 2, we view the probability of each event in Step 1 as the sum of the product of certain probabilities, one of which will be computed analytically. We subsequently refer to this analytical term as a weighting factor.
4. We apply concepts from linear algebra to compute the analytical weighting factor, which corrects the decoding error probabilities obtained from the biased MC simulations (importance sampling) in the equation reached in Step 3.

The first step is to approximate the FER using a union bound equation. For a sufficiently high number of global iterations in the error floor region:

$$FER \approx \sum_{F} \sum_{P} Pr\left\{\mathcal{E}_{F,P}\right\}, \tag{2.4}$$

where F is an (a, b) FOBAS, P is an error pattern (assignment of values of the VNs in the configuration) that satisfies the conditions of F, and $\mathcal{E}_{F,P}$ is the event of decoding error due to an FOBAS F for a specific error pattern P. Note that, unlike for binary LDPC codes, for NB-LDPC codes, FOBAS conditions can be satisfied using more than one set of values for its VNs, and thus a sum over P is needed. The maximum number of (a, b) pairs that can result in detrimental FOBASs is given by Lemma 1.

The second step is to build the overall MR channel output sequence $\mathbf{x}$ (of length n). This sequence is commonly approximated as [51], [56]:

$$\mathbf{x} \approx \mathbf{h} * \mathbf{d} + \mathbf{w}, \tag{2.5}$$

where $\mathbf{h}$ is the transition response sequence of the MR read-head sensitivity, $\mathbf{d}$ is the data sequence, $\mathbf{w}$ is the data dependent noise sequence (data dependency is due to jitter) and $*$ is the convolution sign.

The exponential model [51] of the ideal transition response h of the MR channel read-head sensitivity is given in (2.6) as $h|_{\text{exp}}$:

$$h(t)|_{\text{exp}} = \exp\left[-p_h \left(\frac{t}{T_{50}}\right)^2\right], \tag{2.6}$$

where $t = \ell T$, ℓ is the index of the channel sample, T is the bit duration, T_{50} is the read-head pulse duration at half the amplitude, and p_h is a constant. Since h is decaying with ℓ, it can be sufficiently approximated by a finite number of samples (say τ). Thus, if the ideal MR channel transition response is double-sided, it can be represented as a τ-tuple vector $\mathbf{h} = [h_{-\frac{\tau-1}{2}}\ h_{-\frac{\tau-3}{2}}\ \ldots\ h_{-1}\ h_0\ h_1\ \ldots\ h_{\frac{\tau-3}{2}}\ h_{\frac{\tau-1}{2}}]$ (τ is odd), while if it is right-sided, it can be represented as a τ-tuple vector $\mathbf{h} = [h_0\ h_1\ \ldots\ h_{\tau-1}]$, where $h_\ell = h(\ell T)$.

Remark 5. *The ratio $\frac{T_{50}}{T}$ is called the (normalized) channel density [50], [51], [57]. This* *ratio is important because as it increases, the rate of decay of h decreases, which in turn exacerbates the effect of ISI. The converse is also true.*

A realistic response of the overall MR channel must properly model jitter. Therefore, we specify the term $\mathbf{w}$ in (2.5) as a Gaussian noise sequence $[w_1\ w_2\ \ldots\ w_n]$, which captures both jitter and electronic noise (see [51] and [56]), where w_j has mean 0 and variance σ_j^2. Moreover,

$$\sigma_j^2 \approx \psi_j^2 \sigma_{\text{jitter},j}^2 + \sigma_{\text{elec}}^2, \tag{2.7}$$

where j is the bit index, $1 \leqslant j \leqslant n$, $\sigma_{\text{jitter},j}^2$ and σ_{elec}^2 are the variances of jitter noise and electronic noise, respectively, and ψ_j is a factor representing the effect of jitter propagation on d_j, $\psi_j = f(d_{j-1}, d_j, \frac{\mathrm{d}h(t)}{\mathrm{d}t})$.

One way of modeling the dependence of σ_j on the input data $\mathbf{d}$ is introduced in [23]; by selecting the value of σ_j from two different values depending on whether a transition exists ($d_j \neq d_{j-1}$) or not. Since σ_j depends not only on the transition from d_{j-1} to d_j, but also on the value of d_j, in our method, σ_j takes one of four values. These four values are $\sigma_{0\to0}$, $\sigma_{0\to1}$, $\sigma_{1\to1}$, and $\sigma_{1\to0}$, where $\sigma_{\text{val}_1\to\text{val}_2}$ is the standard deviation of noise at $d_j = \text{val}_2$ given $d_{j-1} = \text{val}_1$. We arbitrarily define $\sigma_{\text{ref}} = \sigma_{1\to0}$ which will be used later.

The third step is to simplify $Pr\left\{\mathcal{E}_{F,P}\right\}$, the probability of a decoding error $\mathcal{E}_{F,P}$ due to F for a specific P, in Equation (2.4) aided by the approximate channel model (see (2.5)):

$$\begin{aligned} Pr\left\{\mathcal{E}_{F,P}\right\} &\approx \sum_{\mathbf{x}} Pr\left\{\mathcal{E}_{F,P}|\mathbf{x}\right\} Pr\left\{\mathbf{x}\right\} \\ &\approx \sum_{\mathbf{w}} \sum_{\mathbf{d}} Pr\left\{\mathcal{E}_{F,P}|\mathbf{w},\mathbf{d}\right\} Pr\left\{\mathbf{w}|\mathbf{d}\right\} Pr\left\{\mathbf{d}\right\}. \end{aligned} \tag{2.8}$$

Let $\mathcal{N} = \{1, 2, \ldots, n\}$. Let $\mathcal{F} = \{j_1, j_2, \ldots, j_{pa}\}$, which is the set of indices of the pa binary modulated bits of F in $\mathbf{d}$, where a is the size of F in symbols $\in \text{GF}(q)$, $q = 2^p$, and all the indices j_i, $1 \leqslant i \leqslant pa$, are indexing the strings after the interleaving step and before the deinterleaving step. Then, $\mathbf{d}^{\mathcal{F}}$ is the result of modulating (e.g., into values in $\{-1, +1\}$ for

BPSK) the binary representation of the codeword symbols of F, i.e., $\mathbf{d}^{\mathcal{F}} = [d_{j_1}\ d_{j_2}\ \ldots\ d_{j_{pa}}]$. Similarly, $\mathbf{w}^{\mathcal{F}}$ is the subsequence of $\mathbf{w}$ indexed by the indices in the set $\mathcal{F}$, while $\mathbf{w}^{\mathcal{N}\setminus\mathcal{F}}$ is the subsequence of $\mathbf{w}$ indexed by all the indices in $\mathcal{N}$ except the indices in $\mathcal{F}$.

In order to simplify (2.8), we separate the noise sequence $\mathbf{w}$ into two parts, $\mathbf{w}^{\mathcal{F}}$ and $\mathbf{w}^{\mathcal{N}\setminus\mathcal{F}}$. As a result, the probability of decoding error due to F, presented in (2.8), can be approximated as follows:

$$Pr\left\{\mathcal{E}_{F,P}\right\} \approx \sum_{\mathbf{w}^{\mathcal{F}}} \sum_{\mathbf{w}^{\mathcal{N}\setminus\mathcal{F}}} \sum_{\mathbf{d}} Pr\left\{\mathcal{E}_{F,P}|\mathbf{w}^{\mathcal{F}}, \mathbf{w}^{\mathcal{N}\setminus\mathcal{F}}, \mathbf{d}\right\} Pr\left\{\mathbf{w}^{\mathcal{F}}|\mathbf{d}\right\} Pr\left\{\mathbf{w}^{\mathcal{N}\setminus\mathcal{F}}|\mathbf{d}\right\} Pr\left\{\mathbf{d}\right\}, \quad (2.9)$$

where noise samples in the random sequence $\mathbf{w}$ given $\mathbf{d}$ are assumed to be independent.

The fourth step is to compute $Pr\left\{\mathbf{w}^{\mathcal{F}}|\mathbf{d}\right\}$, which is our analytical weighting factor customized for the PR channel. This probability is difficult to compute. Instead, we represent it with another conditional probability of a more tractable random variable.

We now show how to carefully construct this auxiliary random variable. Suppose $\mathbf{d}$ is the transmitted binary data sequence vector. Let $\tilde{\mathbf{d}}$ be a decoded vector deviated from $\mathbf{d}$ because of an error $\mathbf{e}$ due to the FOBAS F, i.e., $\tilde{\mathbf{d}} = \mathbf{d} + \mathbf{e}$. It is sufficient to only consider the entries in $\mathbf{d}$ and $\tilde{\mathbf{d}}$ that correspond to the indices in $\mathcal{F}$, i.e., to only consider $\mathbf{d}^{\mathcal{F}}$ and $\widetilde{\mathbf{d}^{\mathcal{F}}}$. Similarly, we define $\mathbf{x}^{\mathcal{F}}$ (resp., $\mathbf{e}^{\mathcal{F}}$) to be the subsequence vector of $\mathbf{x}$ (resp., $\mathbf{e}$) indexed by the indices in the set $\mathcal{F}$. Let $\alpha_{j_i} = \frac{\sigma_{\text{ref}}}{\sigma_{j_i}}$ be the noise scaling factor. (The importance of such scaling will be illustrated shortly.) Then, $\mathbf{d}^{\mathcal{F}}_{\alpha}$, $\mathbf{x}^{\mathcal{F}}_{\alpha}$, and $\mathbf{e}^{\mathcal{F}}_{\alpha}$ are $\mathbf{d}^{\mathcal{F}}$, $\mathbf{x}^{\mathcal{F}}$, and $\mathbf{e}^{\mathcal{F}}$ after scaling each entry j_i by α_{j_i} (see Fig. 2.4):

$$\begin{aligned} \mathbf{d}^{\mathcal{F}}_{\alpha} &= [\alpha_{j_1} d_{j_1}\ \alpha_{j_2} d_{j_2}\ \ldots\ \alpha_{j_{pa}} d_{j_{pa}}], \\ \mathbf{x}^{\mathcal{F}}_{\alpha} &= [\alpha_{j_1} x_{j_1}\ \alpha_{j_2} x_{j_2}\ \ldots\ \alpha_{j_{pa}} x_{j_{pa}}], \end{aligned} \quad (2.10)$$

$$\mathbf{e}^{\mathcal{F}}_{\alpha} = \widetilde{\mathbf{d}^{\mathcal{F}}_{\alpha}} - \mathbf{d}^{\mathcal{F}}_{\alpha} = [\alpha_{j_1} e_{j_1}\ \alpha_{j_2} e_{j_2}\ \ldots\ \alpha_{j_{pa}} e_{j_{pa}}], \quad (2.11)$$

where $\widetilde{\mathbf{d}_\alpha^\mathcal{F}}$ represents $\widetilde{\mathbf{d}^\mathcal{F}}$ after scaling each entry j_i by α_{j_i}. Since $\widetilde{\mathbf{d}^\mathcal{F}}$ critically depends on P in (2.4), $\mathbf{e}^\mathcal{F}$ also critically depends on P. Based on (2.5) and assuming $\mathbf{h}$ to be right-sided for simplicity, we note that $x_{j_i} = h_0 d_{j_i} + h_1 d_{j_i-1} + \cdots + h_{\tau-1} d_{j_i-(\tau-1)} + w_{j_i}$. Additionally, we define the vector $\mathbf{r}_\alpha^\mathcal{F}$, which captures both ISI and noise (see Fig. 2.4 for more illustration), as follows:

$$\mathbf{r}_\alpha^\mathcal{F} = \mathbf{x}_\alpha^\mathcal{F} - \mathbf{d}_\alpha^\mathcal{F} = [\alpha_{j_1} r_{j_1} \ \alpha_{j_2} r_{j_2} \ \dots \ \alpha_{j_{pa}} r_{j_{pa}}]. \tag{2.12}$$

Let $\beta_{j_i} = h_0 d_{j_i} + h_1 d_{j_i-1} + \cdots + h_{\tau-1} d_{j_i-(\tau-1)} - d_{j_i}$. Thus,

$$\alpha_{j_i} r_{j_i} = \alpha_{j_i} x_{j_i} - \alpha_{j_i} d_{j_i} = \alpha_{j_i} \beta_{j_i} + \alpha_{j_i} w_{j_i}. \tag{2.13}$$

Here, we aim to project the vector $\mathbf{r}_\alpha^\mathcal{F}$ onto the vector $\mathbf{e}_\alpha^\mathcal{F}$ in the pa dimensional space. Mathematically, this projection can only be correct if all the pa dimensions are treated equally. In order to treat these dimensions equally, it suffices to appropriately scale each dimension i by α_{j_i} specified above, to unify the variances accross all the dimensions to be σ^2_{ref}, as done in [23]. Notice that scaling each dimension i by α_{j_i} converts $\mathbf{d}^\mathcal{F}$, $\mathbf{e}^\mathcal{F}$, and $\mathbf{x}^\mathcal{F}$ into $\mathbf{d}_\alpha^\mathcal{F}$, $\mathbf{e}_\alpha^\mathcal{F}$, and $\mathbf{x}_\alpha^\mathcal{F}$ that are defined above. The projection process is illustrated in Fig. 2.4. Using linear algebraic concepts and aided by (2.11), (2.12), and (2.13), the result of this projection is:

$$\mathbf{z}_{r,\alpha} = \left(\mathbf{r}_\alpha^\mathcal{F} \cdot \frac{\mathbf{e}_\alpha^\mathcal{F}}{\|\mathbf{e}_\alpha^\mathcal{F}\|_2} \right) \frac{\mathbf{e}_\alpha^\mathcal{F}}{\|\mathbf{e}_\alpha^\mathcal{F}\|_2}. \tag{2.14}$$

We now present the new quantity $S_{F,P}$ for a given $\mathbf{d}$, which will replace $\mathbf{w}^\mathcal{F}$ and will simplify (2.9):

$$\begin{aligned} S_{F,P} &= \mathbf{r}_\alpha^\mathcal{F} \cdot \frac{\mathbf{e}_\alpha^\mathcal{F}}{\|\mathbf{e}_\alpha^\mathcal{F}\|_2^2} \\ &= \frac{1}{\|\mathbf{e}_\alpha^\mathcal{F}\|_2^2} (\alpha_{j_1}^2 e_{j_1} \beta_{j_1} + \alpha_{j_2}^2 e_{j_2} \beta_{j_2} + \cdots + \alpha_{j_{pa}}^2 e_{j_{pa}} \beta_{j_{pa}} \\ &+ \alpha_{j_1}^2 e_{j_1} w_{j_1} + \alpha_{j_2}^2 e_{j_2} w_{j_2} + \cdots + \alpha_{j_{pa}}^2 e_{j_{pa}} w_{j_{pa}}). \end{aligned} \tag{2.15}$$

Thus, $S_{F,P}$ given $\mathbf{d}$ is a Gaussian random variable with **non-zero mean** $\mu_{S_{F,P}}$ and variance $\sigma^2_{S_{F,P}}$:

$$\mu_{S_{F,P}} = \frac{1}{\|\mathbf{e}^{\mathcal{F}}_{\alpha}\|_2^2}(\alpha^2_{j_1} e_{j_1}\beta_{j_1} + \alpha^2_{j_2} e_{j_2}\beta_{j_2} + \cdots + \alpha^2_{j_{pa}} e_{j_{pa}}\beta_{j_{pa}}), \tag{2.16}$$

$$\begin{aligned}\sigma^2_{S_{F,P}} &= \frac{1}{\|\mathbf{e}^{\mathcal{F}}_{\alpha}\|_2^4}(\alpha^2_{j_1} e^2_{j_1} + \alpha^2_{j_2} e^2_{j_2} + \cdots + \alpha^2_{j_{pa}} e^2_{j_{pa}})\sigma^2_{\text{ref}} \\ &= \frac{\sigma^2_{\text{ref}}}{\|\mathbf{e}^{\mathcal{F}}_{\alpha}\|_2^2}.\end{aligned} \tag{2.17}$$

Note that $S_{F,P}$ given $\mathbf{d}$ does not depend on the entire transmitted data sequence $\mathbf{d}$ since only a subset of $\mathbf{d}$ affects it. Let $\mathcal{F}_S$ be the set of indices of the modulated bits in this subset:

$$\mathcal{F}_S = \{j_1 - (\tau - 1), j_1 - (\tau - 2), \ldots, j_1, j_2 - (\tau - 1), j_2 - (\tau - 2), \ldots, j_2, \ldots, j_{pa}\}.$$

Thus, the probability density function (PDF) of $S_{F,P}$ given $\mathbf{d}$, plotted in Fig. 2.5, is:

$$\begin{aligned} f\{S_{F,P}|\mathbf{d}\} &= f\{S_{F,P}|\mathbf{d}^{\mathcal{F}_S}\} \\ &= \frac{1}{\sqrt{2\pi}\sigma_{S_{F,P}}}\exp\left[-\frac{\left(S_{F,P} - \mu_{S_{F,P}}\right)^2}{2\sigma^2_{S_{F,P}}}\right],\end{aligned} \tag{2.18}$$

where $\mu_{S_{F,P}}$ and $\sigma_{S_{F,P}}$ are given in (2.16) and (2.17).

Substituting (2.15) in (2.14) gives the projection outcome $\mathbf{z}_{r,\alpha} = S_{F,P}\mathbf{e}^{\mathcal{F}}_{\alpha}$ (see Fig. 2.4). Using different values of $S_{F,P}$ given $\mathbf{d}$, we move along the vector $\mathbf{e}^{\mathcal{F}}_{\alpha}$ and simulate all the blocks of the receiver. If a decoding error due to F occurs, we compute the analytical weighting factor which is the probability $Pr\left\{S_{F,P}|\mathbf{d}^{\mathcal{F}_S}\right\}$, obtained from (2.18). Thus, we have properly rescaled the contribution of this particular decoding error to the overall FER. With the bias proposed in our prediction method, the simulation time needed is reduced by multiple orders of magnitude compared with the traditional MC simulation.

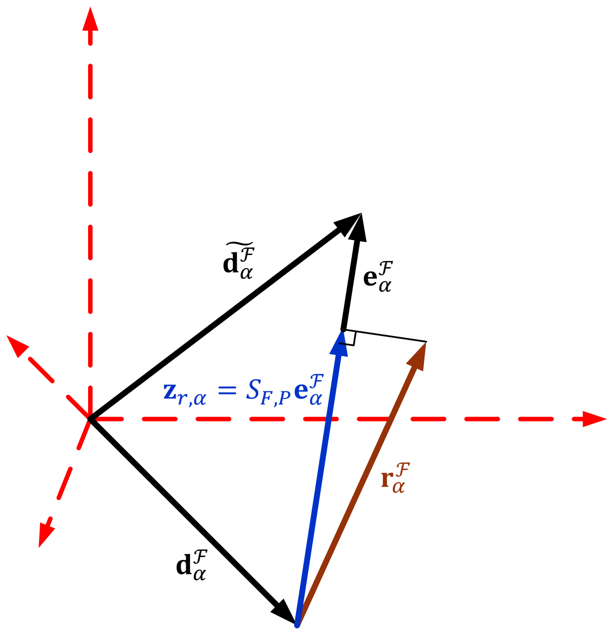

$\widetilde{\mathbf{d}_\alpha^{\mathcal{F}}}$
$\mathbf{e}_\alpha^{\mathcal{F}}$
$\mathbf{z}_{r,\alpha} = S_{F,P}\mathbf{e}_\alpha^{\mathcal{F}}$
$\mathbf{r}_\alpha^{\mathcal{F}}$
$\mathbf{d}_\alpha^{\mathcal{F}}$

Figure 2.4: The process of projecting $\mathbf{r}_\alpha^{\mathcal{F}}$ onto $\mathbf{e}_\alpha^{\mathcal{F}}$ in the pa dimensional space.

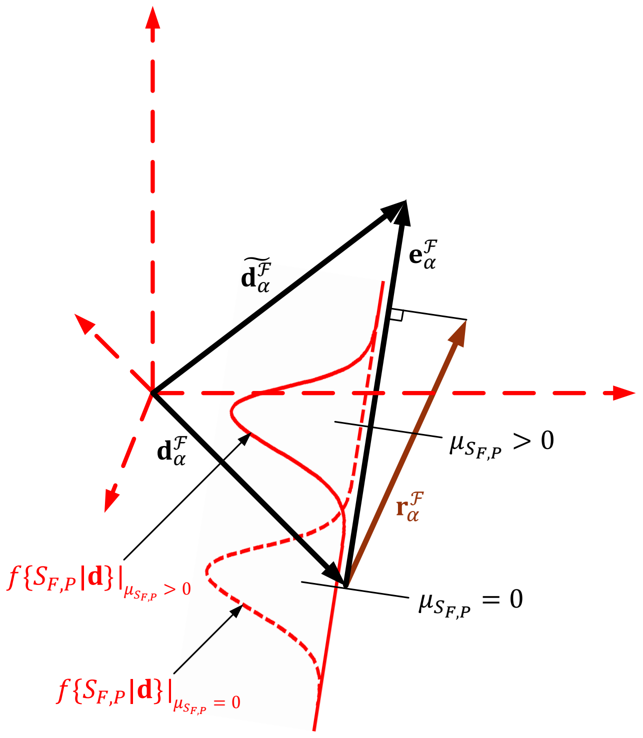

$\widetilde{\mathbf{d}_\alpha^{\mathcal{F}}}$
$\mathbf{e}_\alpha^{\mathcal{F}}$
$\mu_{S_{F,P}} > 0$
$\mathbf{d}_\alpha^{\mathcal{F}}$
$\mathbf{r}_\alpha^{\mathcal{F}}$
$f\{S_{F,P}|\mathbf{d}\}|_{\mu_{S_{F,P}}>0}$
$\mu_{S_{F,P}} = 0$
$f\{S_{F,P}|\mathbf{d}\}|_{\mu_{S_{F,P}}=0}$

Figure 2.5: The effect of the non-zero mean ($\mu_{S_{F,P}} > 0$) on the analytical term $Pr\left\{S_{F,P}|\mathbf{d}^{\mathcal{F}_S}\right\}$. Note how the projection point of $\mathbf{r}_\alpha^{\mathcal{F}}$ over $\mathbf{e}_\alpha^{\mathcal{F}}$ is closer to the mean of the associated distribution when $\mu_{S_{F,P}}$ is positive, and is much further out in the tail of the associated distribution when $\mu_{S_{F,P}}$ is zero.

Accounting for the effect of the ISI on the mean $\mu_{S_{F,P}}$ of $S_{F,P}$ given $\mathbf{d}$ improves the accuracy of the error floor estimate. The reason is that the ISI, which directly affects $\mu_{S_{F,P}}$ as shown in (2.16), is a fundamental source of error in PR channels. Additionally, the analytically-computed term, $Pr\left\{S_{F,P}|\mathbf{d}^{\mathcal{F}_S}\right\}$, varies significantly with the variation of $\mu_{S_{F,P}}$ as shown in Fig. 2.5. Notice that if the ISI is ignored, β_{j_i} is always 0, and from (2.16), $\mu_{S_{F,P}}$ will also be 0 (as in [23]). As simulation results will confirm, having $\mu_{S_{F,P}} = 0$ lowers the weighting factors associated with decoding error events, which in turn underestimates the overall FER.

Finally, we derive the final approximate FER that includes our analytical weighting factor which captures MR channel effects. We replace $\mathbf{w}^{\mathcal{F}}|\mathbf{d}$ with $S_{F,P}|\mathbf{d}$ and consequently with $S_{F,P}|\mathbf{d}^{\mathcal{F}_S}$ (as illustrated above) in (2.9) to reach:

$$Pr\left\{\mathcal{E}_{F,P}\right\} \approx \sum_{S_{F,P}} \sum_{\mathbf{w}^{\mathcal{N}\setminus\mathcal{F}}} \sum_{\mathbf{d}} Pr\left\{\mathcal{E}_{F,P}|S_{F,P}, \mathbf{w}^{\mathcal{N}\setminus\mathcal{F}}, \mathbf{d}\right\} Pr\left\{S_{F,P}|\mathbf{d}^{\mathcal{F}_S}\right\} Pr\left\{\mathbf{w}^{\mathcal{N}\setminus\mathcal{F}}|\mathbf{d}\right\} Pr\left\{\mathbf{d}\right\}. \tag{2.19}$$

Substituting (2.19) in (2.4) results in the final approximate frame error rate:

$$FER \approx \sum_{F} \sum_{P} \sum_{S_{F,P}} \sum_{\mathbf{w}^{\mathcal{N}\setminus\mathcal{F}}} \sum_{\mathbf{d}} Pr\left\{\mathcal{E}_{F,P}|S_{F,P}, \mathbf{w}^{\mathcal{N}\setminus\mathcal{F}}, \mathbf{d}\right\} Pr\left\{S_{F,P}|\mathbf{d}^{\mathcal{F}_S}\right\} Pr\left\{\mathbf{w}^{\mathcal{N}\setminus\mathcal{F}}|\mathbf{d}\right\} Pr\left\{\mathbf{d}\right\}. \tag{2.20}$$

2.3.2 The Algorithm and Simulation Results

In lieu of generating MR channel outputs directly, we compute a biased channel output and the equivalent analytical weighting factor to accelerate MC simulations. (This channel output is said to be biased since we deliberately bias it via channel effects (ISI and noise) towards an error due to the FOBAS of interest at the decoder output.) The MR channel equivalent block is shown in Fig. 2.6, and the remainder of the system is assumed to be as in Fig. 2.1. Note that to compute the biased channel output, the vector $\mathbf{z}_{r,padded}$ is added over

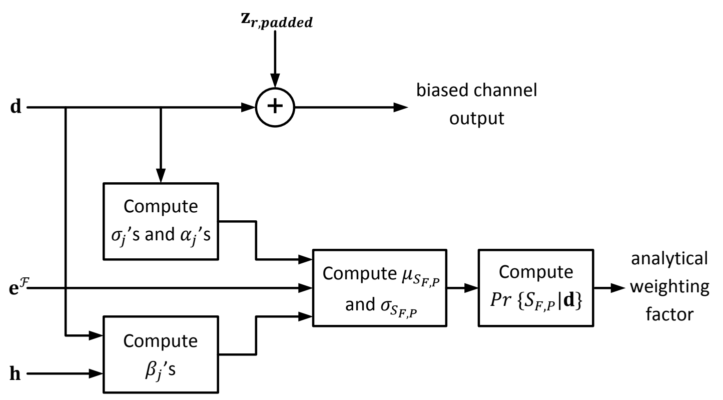

$\mathbf{z}_{r,padded}$
$\mathbf{d}$
biased channel output
Compute σ_j's and α_j's
$\mathbf{e}^{\mathcal{F}}$
Compute $\mu_{S_{F,P}}$ and $\sigma_{S_{F,P}}$
Compute $Pr\{S_{F,P}|\mathbf{d}\}$
analytical weighting factor
Compute β_j's
$\mathbf{h}$

Figure 2.6: MR channel equivalent block of the error floor prediction method.

the data sequence $\mathbf{d}$, where the entries of the vector $\mathbf{z}_{r,padded}$ indexed by $\mathcal{F}$ are described by the vector $\mathbf{z}_r = S_{F,P}\mathbf{e}^{\mathcal{F}}$ (renormalization of $\mathbf{z}_{r,\alpha}$), while all other entries are zeros (or they are the entries indexed by $\mathcal{N} \setminus \mathcal{F}$ in $\mathbf{r} = \mathbf{x} - \mathbf{d}$).

We draw samples from a certain range of $S_{F,P}$ given $\mathbf{d}$ and the probabilities of these samples are obtained from the distribution in (2.18). This range is chosen a priori such that the errors can be generated at the decoder output and such that the value of the analytical term $Pr\left\{S_{F,P}|\mathbf{d}^{\mathcal{F}_S}\right\}$ is not extremely low in order to keep the computations statistically significant (see also [23]). In our case, such range depends not only on the value of $\sigma_{S_{F,P}}$ but also on the value of $\mu_{S_{F,P}}$. An example illustrating this choice will be given shortly.

Remark 6. *Only positive values of $\mu_{S_{F,P}}$ matter (as shown in Fig. 2.5) since negative values result in constructive ISI, which in turn implies much smaller probability of decoding error due to the FOBAS F.*

Algorithm 1 summarizes the steps we use in our error floor prediction method. We use 10 global iterations for all the simulations in this subsection.

We use different NB-PB-LDPC codes, generated according to [8] and [9], to verify the effectiveness of our prediction method. Code 2.1 (resp., Code 2.2) is the code presented in Example 1 (resp., Example 2). Code 2.3 is an NB-PB-LDPC code with $n = 884$ bits, $R \approx 0.82$, $q = 4$, and $\gamma = 3$. Finally, Code 2.4 is an NB-PB-LDPC code with $n = 8178$ bits, $R \approx 0.86$, $q = 8$, and $\gamma = 4$.

Algorithm 1 Error Floor Prediction of Regular NB-LDPC Codes over PR Channels

```
1: Inputs: Tanner graph $G$ of the code, ideal read-head transition response vector $\mathbf{h}$, and SNR range in the error floor region.
2: Set S_max and d_max as the values that specify when to stop looping.
3: Determine $\mathcal{Z}$, the set of all dominant FOBASs, by simulating a representative SNR point. (Combinatorial techniques, e.g., [6], can also help to specify $\mathcal{Z}$ based on the code structure.)
4: for every SNR value $\in$ the SNR range do
5:     for every FOBAS $F \in \mathcal{Z}$ do
6:         From Step 3, determine the set of error patterns $\mathcal{Q}_F$ that satisfy the conditions of $F$.
7:         for every error pattern $P \in \mathcal{Q}_F$ ($\mathbf{e}^{\mathcal{F}}$ is determined through $P$) do
8:             Determine the data sequence $\mathbf{d}$.
9:             Compute $\sigma_j$'s, $\alpha_j$'s, and $\beta_j$'s given $\mathbf{d}$, $\mathbf{e}^{\mathcal{F}}$, $\mathbf{h}$, and SNR value.
10:            Compute $\mu_{S_{F,P}}$ and $\sigma_{S_{F,P}}$ via $\alpha_j$'s, $\beta_j$'s, and $\sigma_{\text{ref}}$.
11:            Determine the value of $S_{F,P}$ based on the computations in 10.
12:            Add the vector $\mathbf{z}_{r,padded}$ over $\mathbf{d}$ to get the biased channel output.
13:            Simulate the whole receiver (CTF, DFIR, detector, and decoder).
14:            if a decoding error due to $F$ occurs then
15:                Calculate the analytical term $Pr\left\{S_{F,P}|\mathbf{d}^{\mathcal{F}_S}\right\}$ from (2.18).
16:            end if.
17:            If S_max is not reached, go to Step 11 to pick another value of $S_{F,P}$.
18:            If d_max is not reached, go to Step 8 to pick another data sequence $\mathbf{d}$.
19:            Compute all the non-analytical terms in (2.19) from the simulation setup and results.
20:        end for
21:    end for
22:    Sum over all the values computed above to get the estimated FER according to (2.20).
23: end for
24: Output: Estimated FER vector for the given SNR range.
```

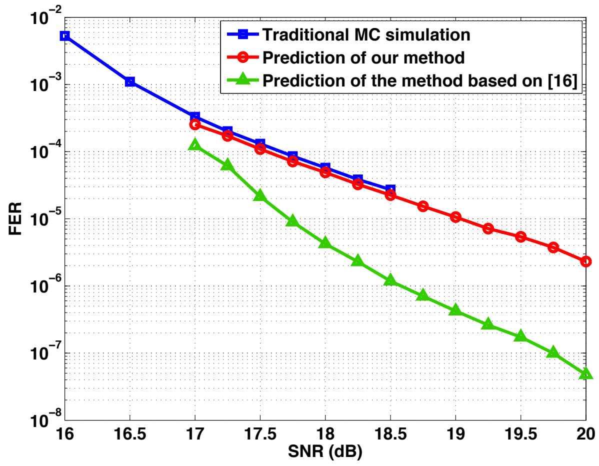


Figure 2.7: Error floor prediction results of Code 2.3.

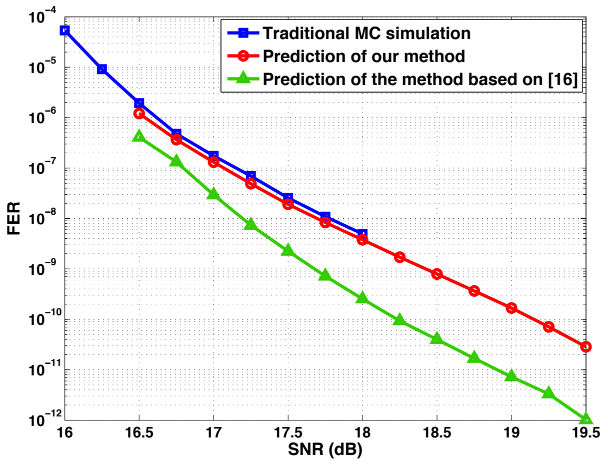


Figure 2.8: Error floor prediction results of Code 2.1.

To perform relevant comparison against the method in [23], we generalize this method for any PR target, change the error objects to FOBASs, and account for the change in the channel density via multiplying by an empirical weighting factor. The PR target used for simulations in this subsection is the default taget, which is [8 14 2]. We tested our prediction method using other PR targets, and our results were consistent with what we present in this subsection. For Figures 2.7, 2.8, and 2.9, the channel density used is 1.4. Figures 2.7, 2.8, and 2.9 show that the error floor estimate obtained using this modified method based on [23] is within 1.5 orders of magnitude from the traditional MC simulation compared with 0.2 of an order of magnitude from the traditional MC simulation achieved by our method. Moreover, the figures show that our error floor estimate correctly captures the error floor slope of the traditional MC simulation for all the simulated codes. Note that, consistent with the analysis in the previous subsection, the prediction method based on [23] underestimates the overall FER.

Example 4. *When we apply our error floor prediction method to Code 2.1, the representative SNR point we simulate is* 16.75 *dB (see also Fig. 2.8). The set* $\mathcal{Z}$ *of the dominant objects consists of the FOBASs* $(6,2)$, $(6,3)$, *and* $(8,2)$. *The suitable range of* $S_{F,P}$ *is between* 0.4 *and* 0.7. *This range can be chosen using a small training set. At* 18 *dB, the error floor prediction method we propose is nearly* 1000 *times faster than the traditional MC simulation.*

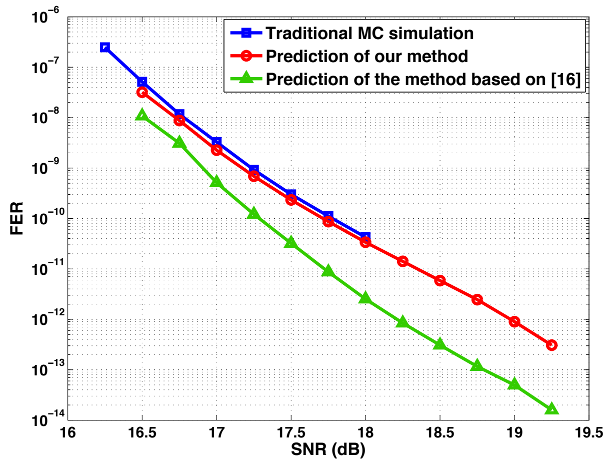


Figure 2.9: Error floor prediction results of Code 2.4.

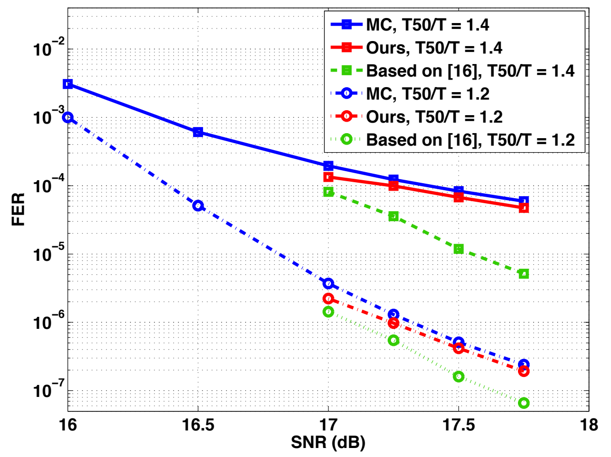


Figure 2.10: Error floor prediction results of Code 2.2 at different values of the MR channel density using the proposed prediction method and the method based on [23].

Fig. 2.10 captures the main difference between the two prediction methods – accounting for the MR channel density $\frac{T_{50}}{T}$. Code 2.2 is simulated over two different PR channels, with channel densities set to 1.4 and 1.2, respectively. Error floor prediction methods are applied to both channels. It is clear that increasing the channel density, i.e., more ISI, increases the gap between the estimate of the modified method based on [23] and the traditional MC simulation while it does not affect the small gap between the estimate of our method and the traditional MC simulation. Fig. 2.10 also shows how our method correctly captures the slope of the error floor performance irrespective of the value of the channel density. We performed different simulations confirming the same conclusions for other values of the channel density, which we have omitted for the sake of clarity in the figure.

2.4 Code Optimization for Transmission over PR Channels

After analyzing the error floor of NB-LDPC codes over PR channels, we turn our attention towards optimizing these codes to achieve better error floor performance. Our goal is to provably eliminate detrimental objects from the Tanner graph of a regular NB-LDPC code, which we have identified to be BASs for PR channels. Our removal process carefully modifies edge weights in the Tanner graph so that the resulting code is free of BASs. We discuss the operations on the edge weights that need to be performed in order to remove an object of interest. The complete code optimization framework, in addition to simulation results over various channels, will be discussed in the following two chapters.

2.4.1 Removing Balanced Absorbing Sets

Since our focus in the PR system is on the restricted subclass of ASs, given by BASs, it is sufficient to eliminate only this special class of objects. The object removal can be performed by changing only the edge weights, i.e., without altering the underlying topology of the Tanner graph. This is a particularly desirable feature as it maintains the underlying implementation-friendly code structure and properties. As a result, we preserve the code structure and rate while improving the code performance. Restricting the object removal process to only the class of BASs offers an additional degree of freedom, compared with the case of removing the more general class of ASs, as would be necessary in other applications. We capture this difference in the following lemma. Recall that $g = \left\lfloor \frac{\gamma-1}{2} \right\rfloor$ for a given column weight (VN degree) γ.

Lemma 2. *The minimum number of edge weights to be changed to remove an (a, b) AS is given by:*

$$E_{\text{AS,min}} = g - b_{\text{vn,max}} + 1, \tag{2.21}$$

where $b_{\text{vn,max}}$ is the maximum number of existing unsatisfied CNs per VN in the subgraph of this object, while the minimum number of edge weights to be changed to remove an (a, b) BAS is given by:

$$E_{\text{BAS,min}} = \min\left\{E_{\text{AS,min}}, E_{\text{BU,min}}\right\}, \tag{2.22}$$

$$\textit{where } E_{\text{BU,min}} = \left\lfloor \frac{ag}{2} \right\rfloor - b + 1. \tag{2.23}$$

Proof. To remove an AS, this configuration needs to be converted into a non-AS. To perform this conversion, it suffices to increase the number of unsatisfied CNs to be (just) above $\left\lfloor \frac{\gamma-1}{2} \right\rfloor$ for any given VN in the configuration. Since $b_{\text{vn,max}}$ is the highest number of unsatisfied CNs connected to a single VN, the number of necessary edge weight changes is then minimized if we choose this VN to begin with. Thus,

$$E_{\text{AS,min}} = \left\lfloor \frac{\gamma - 1}{2} \right\rfloor - b_{\text{vn,max}} + 1 = g - b_{\text{vn,max}} + 1,$$

where the last equality is obtained from the definition of g.

On the other hand, to remove a BAS, we may convert it into a non-AS, and we may also convert it into a UBAS.

To convert a BAS into a UBAS, it suffices to increase b to be (just) above $\left\lfloor \frac{ag}{2} \right\rfloor$. Thus, the minimum number of edge weights to be changed will be:

$$E_{\text{BU,min}} = \left\lfloor \frac{ag}{2} \right\rfloor - b + 1.$$

Computing the minimum of (2.21) and (2.23) gives us the minimum number of edge weights to be changed to remove a BAS. □

Remark 7. *The main source of the extra edge weight selections is the new removal option where we allow a BAS to be converted into a UBAS instead of strictly converting it into a*

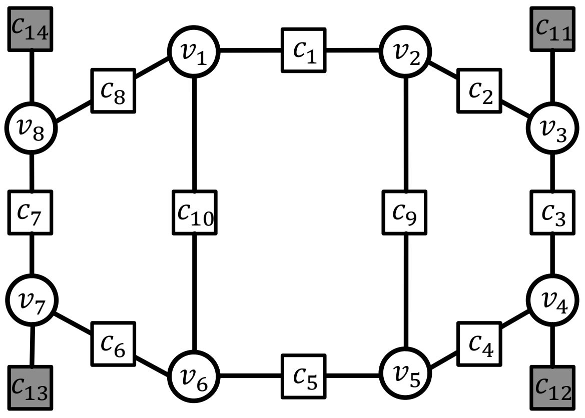


Figure 2.11: An $(8, 4)$ balanced absorbing set, $\gamma = 3$. Circles represent VNs and white (resp., grey) squares represent satisfied (resp., unsatisfied) CNs. Appropriate non-binary edge weights and VN values are assumed.

non-AS. Thus, the introduction of the term $E_{\text{BU,min}}$ of (2.23), which is a result of the new removal option we have for BASs, guarantees additional edge weight options irrespective of whether $E_{\text{BAS,min}} < E_{\text{AS,min}}$ or $E_{\text{BAS,min}} = E_{\text{AS,min}}$ holds.

We illustrate Lemma 2 with the following example.

Example 5. *Fig. 2.11 shows an $(8, 4)$ BAS where $\gamma = 3$ ($g = 1$). Thus, $a = 8$, $b = 4$, and $b_{\text{vn,max}} = 1$. Equations (2.21) and (2.23) from Lemma 2 give $E_{\text{AS,min}} = E_{\text{BU,min}} = 1$, which means $E_{\text{BAS,min}} = 1$. This indicates that the problematic object can be removed by changing only 1 edge weight. From Fig. 2.11, there are $\binom{20}{1}$ selections available to remove the BAS (make it a non-AS or an $(8, 5)$ UBAS), while there are only $\binom{12}{1}$ selections available to convert this structure to a non-AS because the edges connected to c_1, c_5, c_9, and c_{10} do not qualify in this case. Thus, despite that $E_{\text{BAS,min}} = E_{\text{AS,min}}$ in this example, the gain is approximately 67% increase in allowable edge selections when focusing exclusively on BAS removal. The 8 additional edge selections are coming from the term $E_{\text{BU,min}}$ (the additional removal option).*

2.5 Concluding Remarks

In this work, we studied in detail the error floor performance of regular NB-LDPC codes over PR channels. We demonstrated that the error profile over PR channels is different from that over AWGN channels; the existing code optimization techniques previously developed for the AWGN channel transmission are thus not effective. We introduced a restricted class of combinatorial objects, called balanced absorbing sets (BASs), that were identified to be the key contributor to the PR channel error floor. We used these objects to accurately predict the error floor of NB-LDPC codes over PR channels via a comprehensive prediction method. Simulation results revealed that our prediction method estimates the error floor within 0.2 of an order of magnitude from the actual simulation results. Additionally, we have discussed how the exclusive focus on BASs in the code optimization procedure can offer an additional degree of freedom. Future work can include the analysis under symbol-based detection.

Acknowledgement

The majority of the material in this chapter was published in [40]. The work was also presented in part at GLOBECOM 2015 [39]. Additional results for the PR channel are in [47]. The author would like to thank the collaborators in these publications.

CHAPTER 3

Non-Binary LDPC Code Optimization

3.1 Introduction

Due to their excellent performance in other applications, low-density parity-check (LDPC) codes are now actively being considered for modern dense storage devices, such as multi-level Flash and hard disk drives. It is well known that non-binary LDPC (NB-LDPC) codes outperform their binary counterparts [2]. The major flaw of iteratively-decoded LDPC codes (both binary and non-binary) is the error floor phenomenon caused by absorbing sets (ASs) [3], which are detrimental subgraphs in the Tanner graph of the code. While the error floor of binary LDPC codes has been well studied in the literature, e.g., [3], [4], [5], and [6], this problem for NB-LDPC codes is far less explored. First results on the topic of the error floor of NB-LDPC codes include [7], [9], and [10].

The authors of [10] studied a subclass of non-binary ASs (NB ASs), the so-called NB elementary ASs (EASs), and showed that EASs are the root cause of the error floor of NB-LDPC codes over additive white Gaussian noise (AWGN) channels. In Chapter 2, we demonstrated that the nature of the detrimental objects which dominate the error floor region of NB-LDPC codes depends on the channel of interest. As a result, we concluded that using the code optimization techniques developed for AWGN channels is not appropriate for partial-response (PR) channels (the typical 1-D magnetic recording (MR) channels [11]) because of the intrinsic memory the PR system incorporates. While the operational asymmetry in Flash memory systems is well documented [24], [25], the common code-design approach in these systems is still to directly apply LDPC codes optimized for symmetric, AWGN-like channels [26], [27], [28]: "optimize for AWGN, but use on Flash".

In this chapter, we re-visit the existing definitions of combinatorial objects, such as absorbing sets (ASs) and elementary absorbing sets (EASs) [3], [10], that were proved to be useful in the error floor analysis of NB-LDPC codes over AWGN channels. By recognizing that the existing definitions are insufficient to describe the errors for asymmetric channels, we introduce a more finely specified combinatorial object: the *general absorbing set (GAS)*. Additionally, we introduce an important subclass of GASs, which we call *general absorbing sets of type two (GASTs)*. Our NB-LDPC code optimization objective for aggressively asymmetric channels then becomes the removal of GASTs. Through a succinct matrix-theoretic representation, we express a GAST as a set of submatrices, which we call *weight consistency matrices (WCMs)*. By forcing the null spaces of the resultant WCMs to have a particular property, we provably remove detrimental GASTs from the graph representation of the code. We also demonstrate that the WCM definition can be customized to accurately capture the properties of other subclasses of GASTs, e.g., EASs and balanced absorbing sets of type two (BASTs), which are an important subclass of BASs.

Key features of our WCM code optimization framework are that it systematically manipulates the edge weights in the graph representation of a non-binary code while *maintaining all desirable structural properties (node degree, rate, etc.)*, and that it *can be applied to regular NB-LDPC codes used for a wide variety of channels.* Most importantly, this work offers the first theoretical framework for the analysis and design of NB-LDPC codes over realistic storage channels with asymmetry, e.g., the normal-Laplace mixture (NLM) Flash channel [24]. We show the effectiveness of the WCM framework over many channels with different characteristics. We present results for the NLM channel, the Cai Haratsch Mutlu Mai (CHMM) Flash channel [25], the PR channel [11], [12], and the AWGN channel. Over all these channels, the codes optimized using the WCM framework outperform unoptimized codes by a minimum of 1, and up to nearly 2 orders of magnitude in the uncorrectable bit error rate (UBER) [49] or the frame error rate (FER). Furthermore, over asymmetric channels, the codes optimized using the WCM framework also outperform the codes optimized for symmetric channels.

The rest of the chapter is organized as follows. In Section 3.2, we motivate the need for new definitions, and introduce GASs and GASTs. The combinatorial properties of GASTs along with WCMs and how to use WCMs to remove different detrimental objects are presented in Section 3.3. Aided by the theoretical analysis in Section 3.3, the WCM code optimization framework is then proposed in Section 3.4. In Section 3.5, we introduce simulation results over different channels, demonstrating the performance gains offered by the WCM framework. The chapter ends with concluding remarks in Section 3.6.

3.2 New Objects: GASs and GASTs

3.2.1 Motivating Examples

Consider the Tanner graph of an NB-LDPC code. Previous work in [10] (see also [3] and [58]) introduced and studied the following object: an (a, b) non-binary absorbing set (NB AS), with a variable nodes (VNs), b unsatisfied check nodes (CNs), and with each VN having more satisfied than unsatisfied neighboring CNs. No explicit classification with respect to satisfied/unsatisfied CNs of different degrees was made. This in itself was not an issue for symmetric channels, and in fact techniques focusing on the removal of EASs were demonstrated to be very effective [10] for such channels. We recall that an **elementary AS (EAS)** is an AS with all satisfied CNs having degree 2 and all unsatisfied CNs having degree 1 [3], [10]. The following two examples motivate the need for a more refined description of ASs. Example 6 shows that non-elementary ASs are indeed problematic in the case of non-canonical (e.g., asymmetric) channels, while Example 7 illustrates the subtle issue arising from grouping the objects of interest only by their a and b parameters. In all AS figures, circles represent VNs, and white (resp., grey) squares represent satisfied (resp., unsatisfied) CNs. Moreover, in all AS figures, appropriate non-binary edge weights and VN values are assumed.

Example 6. *Consider Code 3.1: a non-binary protograph-based LDPC (NB-PB-LDPC) code [8, 9] defined over GF(4), with block length =* 3,996 *bits, rate* ≈ 0.89*, and column weight* $= 3$*. We simulate Code 3.1 over AWGN, NLM [24], and CHMM [25] channels*[1]*. The simulations reveal that the* $(4, 3)$ *and the* $(6, 2)$ *non-elementary ASs, shown in Figures 3.1(a) and 3.1(b), respectively, result collectively in only about* 2% *of the errors in the error floor region over the AWGN channel. Contrarily, the* $(4, 3)$ *(resp.,* $(6, 2)$*) AS results in nearly* 15% *(resp.,* 12%*) of the errors in the error floor region over the NLM (resp., CHMM) channel (see also Tables 3.1 and 3.3 in Section 3.5). Intriguingly, in the error floor region for the AWGN channel, we recognize that some of the received strings at the input to the decoder incorporate* $(4, 3)$ *and* $(6, 2)$ *errors, along with few other random errors. However, because of the limited magnitudes of the errors in the AWGN case, the decoder typically resolves such non-elementary errors. For example, suppose that a* $(4, 3)$ *error occurs at the decoder input. In the AWGN case, the unsatisfied CN* c_5 *is typically capable of correcting the limited-magnitude errors at* v_2 *and* v_4 *via its messages, which will eventually result in resolving the entire AS error after few decoder iterations. In contrast, the high error magnitudes (a consequence of channel asymmetry) in the case of the NLM channel can make* c_5 *unable to correct the errors at* v_2 *and* v_4 *even after* 200 *iterations, resulting in an AS error that has a degree-*2 *unsatisfied CN (non-elementary AS error) at the decoder output.*

Asymmetric channels can result in excessively high error magnitudes preventing unsatisfied CNs that have degree > 1 from correcting VN errors in an AS. Note that errors of high magnitudes can also happen in the case of PR channels due to channel memory (see Chapter 2). In conclusion, for non-canonical channels, configurations that are not neces-

[1]More details on the simulation setup and the decoder [52] can be found in the experimental section (Section 3.5).

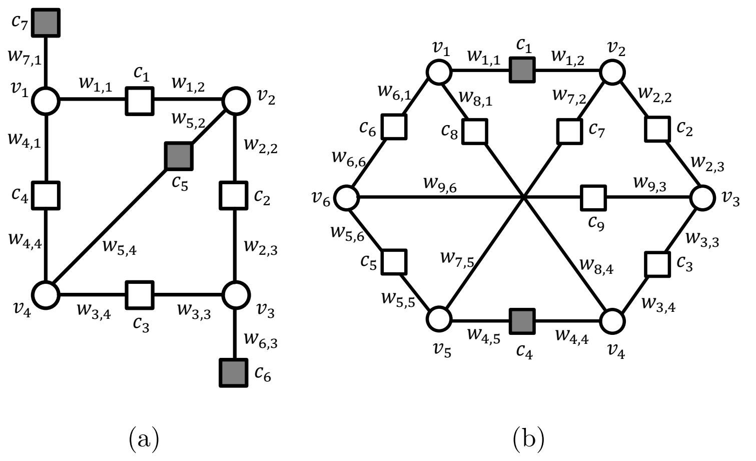


Figure 3.1: (a) A $(4,3)$ non-elementary AS (column weight = 3). (b) A $(6,2)$ non-elementary AS (column weight = 3).

sarily elementary ASs can also be problematic. In this and other cases, code optimization techniques focusing on EASs are ineffective as they are agnostic to a finer classification of the important configurations.

Example 7. *Consider the three NB ASs shown in Fig. 3.2. While they are all classified as a* $(6,2)$ *NB AS, it is clear that they have distinct connectivity properties; configuration (a) is elementary while configurations (b) and (c) are non-elementary (and different).*

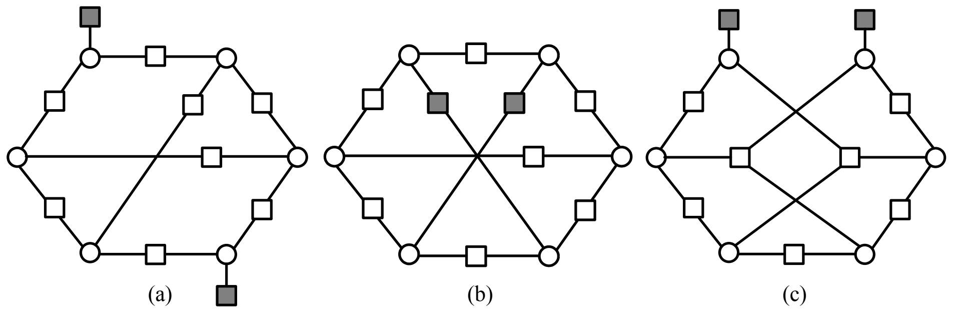


Figure 3.2: Three different $(6,2)$ NB ASs (column weight = 3). Appropriate non-binary edge weights are assumed.

3.2.2 Defining GASs and GASTs

We start off with the definition of a general absorbing set.

Definition 5. *Consider a subgraph induced by a subset $\mathcal{V}$ of VNs in the Tanner graph of an NB-LDPC code. Set all the VNs in $\mathcal{V}$ to values $\in$ GF(q)$\setminus\{0\}$ and set all other VNs to 0. The set $\mathcal{V}$ is said to be an $(a, b, b_2, d_1, d_2, d_3)$ **general absorbing set (GAS)** over GF(q) if and only if the size of $\mathcal{V}$ is a, the number of unsatisfied (resp., degree-2 unsatisfied) CNs connected to $\mathcal{V}$ is b (resp., b_2), the number of degree-1 (resp., 2 and > 2) CNs connected to $\mathcal{V}$ is d_1 (resp., d_2 and d_3), and each VN in $\mathcal{V}$ is connected to strictly more satisfied than unsatisfied neighboring CNs, for some set of VN values.*

In this work, we focus on GF(q) with $q = 2^\lambda$, where λ is a positive integer $\geqslant 2$.

Observe that for a GAS, satisfied CNs are of degree two or higher, and unsatisfied CNs are of degree one or higher ($b \geqslant d_1 + b_2$). This GAS definition explicitly differentiates among the three configurations in Fig. 3.2: configuration (a) is a $(6, 2, 0, 2, 8, 0)$ GAS, configuration (b) is a $(6, 2, 2, 0, 9, 0)$ GAS, and configuration (c) is a $(6, 2, 0, 2, 5, 2)$ GAS.

The description of a GAS depends on having appropriate non-zero non-binary values (labels) associated with its edge weights. It is useful to view the induced subgraph in terms of its unlabeled variant: unlabeled version of a configuration is the configuration with all edge weights set to 1. Thus, we also define the following graph-theoretic object.

Definition 6. *Let $\mathcal{V}$ be a subset of VNs in the unlabeled Tanner graph of an NB-LDPC code. Let $\mathcal{O}$ (resp., $\mathcal{T}$ and $\mathcal{H}$) be the set of degree-1 (resp., 2 and > 2) CNs connected to $\mathcal{V}$. This graphical configuration is an (a, d_1, d_2, d_3) **unlabeled GAS** if it satisfies the following two conditions:*

1. *$|\mathcal{V}| = a$, $|\mathcal{O}| = d_1$, $|\mathcal{T}| = d_2$, and $|\mathcal{H}| = d_3$.*

2. *Each VN in $\mathcal{V}$ is connected to more neighbors in $(\mathcal{T} \cup \mathcal{H})$ than in $\mathcal{O}$.*

Note that an unlabeled GAS is viewed in purely topological terms, and the parameters b and b_2 (unlike for a GAS) are irrelevant.

To understand the second condition in Definition 6, note that among all the CNs connected to the labeled GAS, only degree-1 CNs are guaranteed to be unsatisfied whatever the edge weights are.

We now establish a matrix-theoretic representation of a GAS, building in part on the previous results from [7], [10], and a conceptually connected work from [4]. The null spaces of the corresponding matrices will play the central role in the WCM code optimization framework, as we describe in the next section.

Let $\mathbf{H}$ denote the parity-check matrix of an NB-LDPC code defined over GF(q). Consider an $(a, b, b_2, d_1, d_2, d_3)$ GAS in the Tanner graph of this code. Let $\mathbf{A}$ be the $\ell \times a$ submatrix of $\mathbf{H}$ that consists of $\ell = d_1 + d_2 + d_3$ rows of $\mathbf{H}$, corresponding to the CNs participating in this GAS, and a columns of $\mathbf{H}$, corresponding to the VNs participating in this GAS.

Lemma 3. *An $(a, b, b_2, d_1, d_2, d_3)$ GAS must satisfy:*

- ***Topological conditions:*** *Its unlabeled configuration must satisfy the unlabeled GAS conditions stated in Definition 6.*

- ***Weight conditions:*** *The set is an $(a, b, b_2, d_1, d_2, d_3)$ GAS over GF(q) if and only if there exists an $(\ell - b) \times a$ submatrix $\mathbf{W}$ of column rank $r_{\mathbf{W}} < a$, with elements $w_{i,j}$, $1 \leqslant i \leqslant (\ell - b)$, $1 \leqslant j \leqslant a$, of the GAS adjacency matrix $\mathbf{A}$, that satisfies the following two conditions:*

  1. *Let $\mathcal{N}(\mathbf{W})$ be the null space of the submatrix $\mathbf{W}$, and let $\mathbf{d}_k^{\mathrm{T}}$, $1 \leqslant k \leqslant b$, be the kth row of the matrix $\mathbf{D}$ obtained by removing the rows of $\mathbf{W}$ from $\mathbf{A}$. Let $\mathbf{v}$ be a vector of VN values and $\mathbf{R}$ be an $\ell \times \ell$ permutation matrix. Then,*

$$\exists\ \mathbf{v} = [v_1\ v_2\ \ldots\ v_a]^{\mathrm{T}} \in \mathcal{N}(\mathbf{W}) \textit{ s.t. } v_j \neq 0,\ \forall j \in \{1, 2, \ldots, a\},$$
$$\textit{and } \mathbf{d}_k^{\mathrm{T}}\mathbf{v} = m_k \neq 0,\ \forall k \in \{1, 2, \ldots, b\},\ \mathbf{m} = [m_1\ m_2\ \ldots\ m_b]^{\mathrm{T}},$$
$$\textit{i.e.,}\ \mathbf{RAv} = \begin{bmatrix} \mathbf{W}_{(\ell-b)\times a} \\ \mathbf{D}_{b\times a} \end{bmatrix} \mathbf{v}_{a\times 1} = \begin{bmatrix} \mathbf{0}_{(\ell-b)\times 1} \\ \mathbf{m}_{b\times 1} \end{bmatrix}. \tag{3.1}$$

  2. *Let $d_{k,j}$, $1 \leqslant k \leqslant b$, $1 \leqslant j \leqslant a$, be the elements of the matrix $\mathbf{D}$. Then, $\forall j \in$*

$\{1, 2, \ldots, a\}$,

$$\left(\sum_{i=1}^{\ell-b} F\left(w_{i,j}\right)\right) > \left(\sum_{k=1}^{b} F\left(d_{k,j}\right)\right), \tag{3.2}$$

where $F(\beta) = 0$ if $\beta = 0$, and $F(\beta) = 1$ otherwise.

Computations are performed over GF(q).

Proof. The proof follows from Definition 5. □

In words, $\mathbf{W}$ is the submatrix of satisfied CNs, and $\mathbf{D}$ is the submatrix of unsatisfied CNs. Note that Condition 2 simply ensures that each VN is connected to more satisfied than unsatisfied CNs.

In the rest of this chapter, we study a subclass of GASs, which is defined as follows:

Definition 7. *A GAS that has $d_2 > d_3$ and all the unsatisfied CNs connected to it (if any) $\in (\mathcal{O} \cup \mathcal{T})$ (having either degree 1 or degree 2), is defined as an (a, b, d_1, d_2, d_3)* ***general absorbing set of type two (GAST)****. The word "two" refers to the maximum degree of any unsatisfied CN connected to the set. In a way similar to the unlabeled GAS (see Definition 6), we also define the (a, d_1, d_2, d_3)* ***unlabeled GAST****.*

The reason why we focus on GASTs is that the existence of unsatisfied CNs of degree > 2 in any configuration in the Tanner graph significantly increases the likelihood that the object is not an AS. For example, consider configuration (c) in Fig. 3.2. If either of the two degree-3 CNs is unsatisfied, the resulting object will not be an AS.

In this work, we focus on regular codes, and we let γ denote the column weight (the VN degree) of the code.

Remark 8. *Fig. 3.3 shows how different types of absorbing sets are related. Assuming that the degree of any unsatisfied CN is $\leqslant 2$, different types of known ASs become special cases of GASTs. We summarize this in the following lines:*

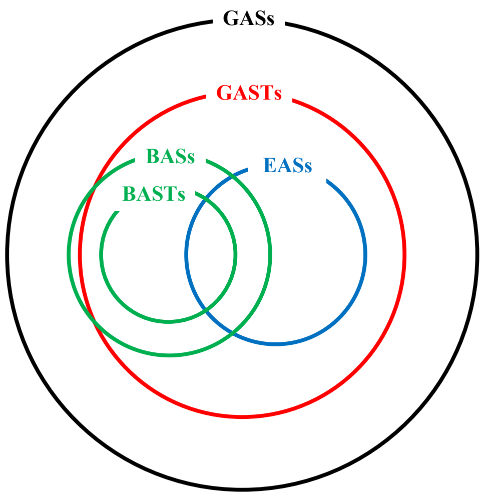


Figure 3.3: A Venn diagram showing the relation between different types of absorbing sets.

- *A GAST with* $b = d_1$ *and* $d_3 = 0$ *is an elementary AS (EAS).*
- *A GAST with* $b > d_1$ *or/and* $d_3 > 0$ *is a non-elementary AS.*
- *A GAST with* $0 \leqslant b \leqslant \left\lfloor \frac{ag}{2} \right\rfloor$ *is a balanced AS (BAS), where* $g = \left\lfloor \frac{\gamma-1}{2} \right\rfloor$.
- *A GAST with* $\left\lfloor \frac{ag}{2} \right\rfloor < b \leqslant ag$ *is an unbalanced AS.*

Balanced and unbalanced ASs were previously introduced in Chapter 2. In particular, balanced ASs (BASs) play a critical role in the context of channels with memory, such as those encountered in magnetic recording applications (e.g., the PR channel).

We devote the next section to establishing a series of properties of GASTs and techniques to remove them from the Tanner graph of the NB-LDPC code.

3.3 Theoretical Analysis of GASTs

3.3.1 Combinatorial Properties of GASTs

The following theorem provides a condition on when a degree-2 CN can be unsatisfied in a GAST configuration which results from an unlabeled GAST configuration operating on the same set of VNs and their neighboring CNs, with proper edge labeling.

Theorem 1. *Consider an (a, d_1, d_2, d_3) unlabeled GAST with $\mathcal{T}$ denoting the set of d_2 degree-2 CNs and with $\mathcal{H}$ denoting the set of d_3 CNs of degree > 2 in this configuration. This unlabeled GAST can result in an (a, b, d_1, d_2, d_3) GAST (with proper edge labeling) that has $b > d_1$ if and only if there exists at least one CN c in $\mathcal{T}$ such that the two neighboring VNs of c (with respect to this unlabeled GAST) each has the property that strictly more than $\left\lceil \frac{\gamma+1}{2} \right\rceil$ of its neighboring CNs belong to $(\mathcal{T} \cup \mathcal{H})$, where γ is the column weight.*

Proof. We prove Theorem 1 by contradiction. We assume that in the unlabeled GAST, there exist no CNs $\in \mathcal{T}$ connecting pairs of VNs that have strictly $> \left\lceil \frac{\gamma+1}{2} \right\rceil$ neighboring CNs $\in (\mathcal{T} \cup \mathcal{H})$ connected to each of them. Thus, for each degree-2 CN, at least one of the connected VNs has exactly $\left\lceil \frac{\gamma+1}{2} \right\rceil$ connected CNs $\in (\mathcal{T} \cup \mathcal{H})$. Note that the number of satisfied CNs connected to any VN in any AS cannot be $< \left\lceil \frac{\gamma+1}{2} \right\rceil$. Also note that these CNs $\in (\mathcal{T} \cup \mathcal{H})$ are the only CNs that can be satisfied because the others $\in \mathcal{O}$ are always unsatisfied.

As a result, if any degree-2 CN is forced to be unsatisfied in the resulting GAST (by the choice of the edge weights), at least one connected VN of the two will have the following bound on the number of satisfied CNs connected to it:

$$\xi_{vn} \leqslant \left\lceil \frac{\gamma+1}{2} \right\rceil - 1 = \left\lceil \frac{\gamma-1}{2} \right\rceil . \tag{3.3}$$

Thus, and since an AS condition is to have $\geqslant \left\lceil \frac{\gamma+1}{2} \right\rceil$ satisfied CNs connected to each VN in the configuration, the configuration under analysis will not become an AS, and thus, will not become a GAST, if any degree-2 CN is unsatisfied. In other words, given the above assumption, all the unsatisfied CNs must be $\in \mathcal{O}$ ($\notin \mathcal{T}$), like EASs, in order that the object is an AS after edge labeling. One check node (at least) violating the above assumption makes it possible to have $b > d_1$ in the resulting GAST. □

Remark 9. *The importance of Theorem 1 is that it provides the necessary topological condition on a GAST to have unsatisfied CNs of degree 2. If this condition is not satisfied, then* *all the unsatisfied CNs of this GAST are of degree* 1*, which makes the removal process of the GAST much easier as we shall see later.*

In this chapter and the next one, the notation "ut" (resp., "et") in the subscript of b refers to the *upper bound on the* (resp., *exact*) maximum number of *degree*-2 unsatisfied CNs.

Theorem 2. *Given an* (a, d_1, d_2, d_3) *unlabeled GAST, the maximum number of unsatisfied CNs,* b_{max}*, in the resulting GAST after edge labeling is upper bounded by:*

$$b_{\text{max}} \leqslant d_1 + b_{\text{ut}}, \text{ where} \tag{3.4}$$

$$b_{\text{ut}} = \left\lfloor \frac{1}{2} \left(a \left\lfloor \frac{\gamma - 1}{2} \right\rfloor - d_1 \right) \right\rfloor. \tag{3.5}$$

Proof. Since degree-1 CNs are always unsatisfied, we can write the bound on b_{max} as follows:

$$b_{\text{max}} \leqslant d_1 + b_{\text{ut}},$$

where b_{ut} is the upper bound on the maximum number of degree-2 unsatisfied CNs ($\in \mathcal{T}$). In the beginning, we mark all the CNs $\in (\mathcal{T} \cup \mathcal{H})$ as satisfied.

To compute b_{ut}, we access all the VNs in the GAST configuration one by one and mark the maximum number of degree-2 CNs that can be unsatisfied simultaneously while the object remains a GAST. As with any AS, the maximum number of unsatisfied CNs connected to a VN in the GAST is $\left\lfloor \frac{\gamma-1}{2} \right\rfloor$ (which is g). Thus, the maximum number of additional CNs connected to VN j, $1 \leqslant j \leqslant a$, that can be marked as unsatisfied is $b_{\text{ut},j} = \left\lfloor \frac{\gamma-1}{2} \right\rfloor - b_{\text{up},j}$, where $b_{\text{up},j}$ is the number of already-unsatisfied CNs connected to VN j **updated** by what has been done for all the VNs processed prior to VN j.

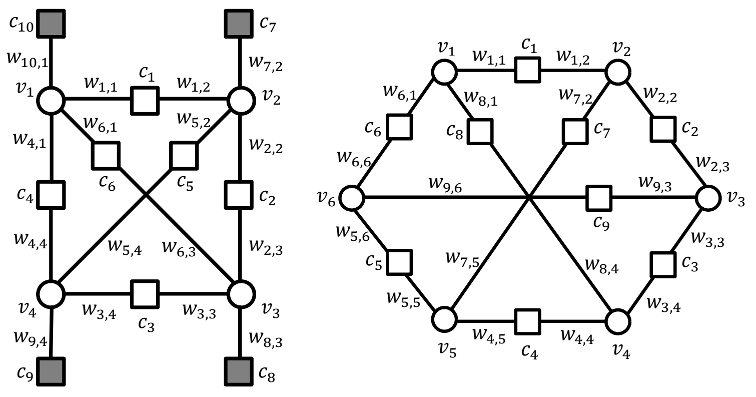
The upper bound b_{ut} is achieved if for each VN that has $> \left\lceil \frac{\gamma+1}{2} \right\rceil$ connected CNs marked as satisfied, there exists another VN connected to it, through some degree-2 satisfied CN, which also has $> \left\lceil \frac{\gamma+1}{2} \right\rceil$ connected CNs marked as satisfied. In other words, the degree-2 CN connecting these two VNs can be marked as unsatisfied while the object remains a GAST. Thus, for some a, d_1, and γ,

$$
\begin{aligned}
b_{\text{ut}} = \sum_{j=1}^{a} b_{\text{ut},j} &= \sum_{j=1}^{a} \left[\left\lfloor \frac{\gamma-1}{2} \right\rfloor - b_{\text{up},j} \right] \\
&= a \left\lfloor \frac{\gamma-1}{2} \right\rfloor - \sum_{j=1}^{a} b_{\text{up},j}. \qquad (3.6)
\end{aligned}
$$

Since $\sum_{j=1}^{a} b_{\text{up},j}$ represents the final number of unsatisfied CNs we will end up with after processing all the VNs in the GAST, it can be concluded that:

$$
\sum_{j=1}^{a} b_{\text{up},j} = d_1 + b_{\text{ut}}. \qquad (3.7)
$$

Substituting (3.7) into (3.6) results in a recursive equation where b_{ut} appears in both the RHS and the LHS. The solution of this equation is (3.5), which completes the proof. □



(a) (b)

Figure 3.4: (a) A $(4, 4, 4, 6, 0)$ GAST, $\gamma = 4$. (b) A $(6, 0, 0, 9, 0)$ GAST, $\gamma = 3$. Unlabeled GASTs are reached by setting all the weights in the configurations to 1.

Example 8. *Consider the* $(4, 4, 6, 0)$ *unlabeled GAST (*$\gamma = 4$*) shown in Fig. 3.4(a). From Theorem 1, irrespective of the edge labeling that converts this unlabeled GAST to a GAST, the resultant GAST cannot have unsatisfied CNs of degree* 2*. From (3.5),* $b_{\text{ut}} = 0$ *and thus,* *from (3.4),* $b_{\max} = b = d_1 = 4$*. Hence, this GAST can only be an EAS. In contrast, the* $(6, 0, 9, 0)$ *unlabeled GAST (*$\gamma = 3$*), shown in Fig. 3.4(b), has the following property. When this unlabeled GAST is converted to a GAST for some GF(q), it is possible to have degree-*2 *unsatisfied CNs (from Theorem 1). Also from (3.5),* $b_{\text{ut}} = 3$*, and thus,* $b_{\max} = b_{\text{ut}} = 3 \neq d_1 = 0$.

3.3.2 How to Remove GASTs Using WCMs

The next step is to make use of Theorems 1 and 2 to remove GASTs from the Tanner graph of the code. First, we need a new definition that captures the removal process. The objective is to avoid generating another GAST for the same unlabeled GAST configuration, after making the edge weight changes. Note that throughout the dissertation, the edge weight changes are with respect to the original configuration. Moreover, the original and the new weights are $\in \text{GF}(q)\backslash\{0\}$.

For a given (a, b, d_1, d_2, d_3) GAST, let $\mathcal{Z}$ be the set of all (a, b', d_1, d_2, d_3) GASTs with $d_1 \leqslant b' \leqslant b_{\max}$, which have the same unlabeled GAST configuration as the original (a, b, d_1, d_2, d_3) GAST. Here, $b_{\max}$ is the largest allowable number of unsatisfied CNs for these configurations.

Definition 8. *An* (a, b, d_1, d_2, d_3) ***GAST*** *is said to be* ***removed*** *from the Tanner graph of an NB-LDPC code if and only if the resulting object (after edge weight processing)* $\notin \mathcal{Z}$.

To clarify Definition 8, we discuss again the two configurations in Fig. 3.4. For the $(4, 4, 4, 6, 0)$ GAST ($\gamma = 4$), $b_{\max} = b = d_1 = 4$, which means $\mathcal{Z}$ contains only one object that is the $(4, 4, 4, 6, 0)$ GAST itself. On the other hand, for the $(6, 0, 0, 9, 0)$ GAST ($\gamma = 3$), $b_{\max} = b_{\text{ut}} = 3$, which means $\mathcal{Z} = \{(6, 0, 0, 9, 0), (6, 1, 0, 9, 0), (6, 2, 0, 9, 0), (6, 3, 0, 9, 0)\}$.

For a given GAST, define a matrix $\mathbf{W}^{\text{z}}$ to be the matrix obtained by removing b', $d_1 \leqslant b' \leqslant b_{\max}$, rows corresponding to CNs $\in (\mathcal{O} \cup \mathcal{T})$ from the matrix $\mathbf{A}$, which is the GAST adjacency matrix (see also Lemma 3). These b' CNs can simultaneously be unsatisfied under some edge labeling that produces a GAST which has the same unlabeled GAST as the given

GAST. Let $\mathcal{U}$ be the set of all matrices $\mathbf{W}^{\mathrm{z}}$. Each element $\in \mathcal{Z}$ has one or more matrices $\in \mathcal{U}$. In principle, GASTs can be removed by manipulating the associated matrices $\mathbf{W}^{\mathrm{z}}$ of the set $\mathcal{U}$. However, a more efficient approach is to work with matrices that are each a submatrix of multiple $\mathbf{W}^{\mathrm{z}}$ matrices. In this way, we can remove problematic GASTs while only focusing on a smaller collection of matrices. These new matrices are referred to as the weight consistency matrices (WCMs).

In this chapter and the next one, the notation "z" (resp., "cm") in the superscript of a matrix means that the matrix is *associated with an element in the set* $\mathcal{Z}$ (resp., *a WCM*).

Definition 9. *For a given* (a, b, d_1, d_2, d_3) *GAST and its associated adjacency matrix* $\mathbf{A}$ *and its associated set* $\mathcal{Z}$*, we construct a set of* t *matrices as follows:*

1. *Each matrix* $\mathbf{W}_h^{\mathrm{cm}}$*,* $1 \leqslant h \leqslant t$*, in this set is an* $(\ell - b_h^{\mathrm{cm}}) \times a$ *submatrix,* $d_1 \leqslant b_h^{\mathrm{cm}} \leqslant b_{\max}$*, formed by removing* ***different*** b_h^{cm} *rows from the* $\ell \times a$ *matrix* $\mathbf{A}$ *of the GAST. These* b_h^{cm} *rows to be removed correspond to CNs* $\in (\mathcal{O} \cup \mathcal{T})$ *that can simultaneously be unsatisfied under some edge labeling that produces a GAST which has the same unlabeled GAST as the given GAST.*

2. *Each matrix* $\mathbf{W}^{\mathrm{z}} \in \mathcal{U}$*, for every element* $\in \mathcal{Z}$*, contains at least one element of the resultant set as its submatrix.*

3. *This resultant set has the* ***smallest cardinality****, which is* t*, among all the sets which satisfy Conditions 1 and 2 stated above.*

We refer to the matrices in this set as ***weight consistency matrices (WCMs)****, and to this set itself as* $\mathcal{W}$*.*

Remark 10. *In many cases, all the WCMs* $\in \mathcal{W}$ *are of the same size, which is* $(\ell - b_{\max}) \times a$*, i.e.,* $b_h^{\mathrm{cm}} = b_{\max}$*,* $\forall h$*.*

Now, we provide the theorem that makes use of the WCMs of the GAST to remove the latter.

Theorem 3. *The necessary and sufficient processing needed to remove an* (a, b, d_1, d_2, d_3) *GAST, according to Definition 8, is to change the edge weights such that for every WCM* $\mathbf{W}_h^{\text{cm}} \in \mathcal{W}$*, there does not exist any vector with all its entries* $\neq 0$ *in the null space of that WCM. Mathematically,* $\forall h$*:*

$$\text{If } \mathcal{N}(\mathbf{W}_h^{\text{cm}}) = \text{span}\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{p_h}\}, \text{ then } \nexists\ \mathbf{r} = [r_1\ r_2\ \dots\ r_{p_h}]^{\text{T}} \text{ for}$$
$$\mathbf{v} = r_1\mathbf{x}_1 + r_2\mathbf{x}_2 + \dots + r_{p_h}\mathbf{x}_{p_h} = [v_1\ v_2\ \dots v_a]^{\text{T}} \text{ s.t. } v_j \neq 0,\ \forall j \in \{1, 2, \dots, a\}, \tag{3.8}$$

where p_h *is the dimension of* $\mathcal{N}(\mathbf{W}_h^{\text{cm}})$*. Computations are performed over GF(q).*

Proof. We divide the proof into two parts. First, we prove that breaking the weight conditions stated in (3.1) and (3.2) for any submatrix $\mathbf{W}^{\text{z}}$ is done as stated in (3.8). From Lemma 3, these weight conditions are broken if:

$$\nexists\ \mathbf{v} = [v_1\ v_2\ \dots\ v_a]^{\text{T}} \in \mathcal{N}(\mathbf{W}^{\text{z}}) \text{ s.t. } v_j \neq 0,\ \forall j \in \{1, 2, \dots, a\}. \tag{3.9}$$

Since the set of vectors $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_p\}$ is a basis of $\mathcal{N}(\mathbf{W}^{\text{z}})$, therefore, if there is no linear combination of them over GF(q) that results in $\mathbf{v}$ s.t. $v_j \neq 0$, $\forall j \in \{1, 2, \dots, a\}$, (3.9) is automatically satisfied. In other words, if (3.8) is satisfied, the weight conditions of $\mathbf{W}^{\text{z}}$ are broken.

Second, we prove that breaking such weight conditions for all the WCMs $\in \mathcal{W}$ (the smallest AS submatrices) guarantees the GAST removal. By the definition of WCMs (Definition 9), $\exists\ \mathbf{W}_h^{\text{cm}} \in \mathcal{W}$ as a submatrix, for any matrix $\mathbf{W}^{\text{z}} \in \mathcal{U}$ of size $(\ell - b') \times a$, $d_1 \leqslant b' < b_{\max}$.

Now, recall the following linear algebraic lemma:

$$\text{If we have a matrix } \mathbf{M} = \begin{bmatrix} \mathbf{M}_1 \\ \mathbf{M}_2 \end{bmatrix},$$
$$\text{then, } \mathcal{N}(\mathbf{M}) \subseteq \mathcal{N}(\mathbf{M}_1), \text{ and } \mathcal{N}(\mathbf{M}) \subseteq \mathcal{N}(\mathbf{M}_2). \tag{3.10}$$

Applying this lemma to our case, if $\nexists$ $\mathbf{v} = [v_1 \ v_2 \ \dots \ v_a]^{\mathrm{T}} \in \mathcal{N}(\mathbf{W}_h^{\mathrm{cm}})$ s.t. $v_j \neq 0$, $\forall j$, for every WCM $\in \mathcal{W}$, then this implies that $\nexists$ $\mathbf{v} = [v_1 \ v_2 \ \dots \ v_a]^{\mathrm{T}} \in \mathcal{N}(\mathbf{W}^{\mathrm{z}})$ s.t. $v_j \neq 0$, $\forall j$, for every matrix $\mathbf{W}^{\mathrm{z}}$ of size $(\ell - b') \times a$, $d_1 \leqslant b' \leqslant b_{\max}$. This property ensures the removal of the GAST according to Definition 8. The necessity follows from that every WCM $\in \mathcal{W}$ is itself a matrix $\mathbf{W}^{\mathrm{z}}$ with no submatrices $\in \mathcal{U}$, which completes the proof of Theorem 3. □

Remark 11. *The concepts proposed by Theorem 3 are not only useful for GAST removal, but also for GAST detection. In other words, using Theorem 3, we can detect whether a certain configuration cannot be an (a, b', d_1, d_2, d_3) GAST, $d_1 \leqslant b' \leqslant b_{\max}$, by checking the null spaces of all the WCMs $\in \mathcal{W}$ of that configuration.*

3.3.3 How to Remove Other Detrimental Objects Using WCMs

Recall that an elementary AS (EAS) is an AS with all satisfied CNs having degree 2 and all unsatisfied CNs having degree 1, while a balanced AS (BAS) is an AS with $0 \leqslant b \leqslant \left\lfloor \frac{ag}{2} \right\rfloor$, where $g = \left\lfloor \frac{\gamma - 1}{2} \right\rfloor$. As shown in Fig. 3.3, every EAS is indeed a GAST. On the other hand, few BASs are not GASTs. For the sake of convenience, we will only focus in the rest of the chapter on BASs that are GASTs (unsatisfied CNs are of either degree 1 or degree 2). In a way similar to the GAST, we call a BAS that has the degree of any unsatisfied CN $\leqslant 2$, a **BAS of type two (BAST)**.

Removing EASs (for symmetric channels) or BASTs (for PR channels), as restricted subclasses of GASTs, requires less steps and fewer edge weight changes compared with removing GASTs according to Definition 8. In other words, it is enough for an elementary AS to be converted into a non-elementary AS, and for a balanced AS to be converted into an unbalanced AS. In order to customize the WCM framework for removing such simpler objects (compared with GASTs), the definitions of WCMs should be customized to only capture the objects of interest (which depend on the channel). Such customization secures additional edge weight choices.

In a way similar to the unlabeled GAST, we define the **unlabeled EAS** and the **un-**

labeled BAST. We note that Lemma 3 applies to EASs and to BASTs. Additionally, Theorem 1 is not needed for EASs since the degree of any unsatisfied CN in an EAS is always 1. On the contrary, Theorem 1 applies to BASTs if $\left\lfloor \frac{ag}{2} \right\rfloor > d_1$. This is because if $\left\lfloor \frac{ag}{2} \right\rfloor < d_1$, the object is not a BAST. Moreover, if $\left\lfloor \frac{ag}{2} \right\rfloor = d_1$, any additional unsatisfied CN will convert the object from a BAST into an unbalanced AS (assuming that the object stays as an AS). Note that degree-1 CNs are always unsatisfied.

We modify Theorem 2 to capture EAS and BAST properties in the following Lemma.

Lemma 4. *Given an $(a, d_1, d_2, 0)$ unlabeled EAS, the maximum number of unsatisfied CNs, $b_{\text{e\_max}}$, in the resulting EAS after edge labeling is given by:*

$$b_{\text{e\_max}} = b = d_1. \tag{3.11}$$

Moreover, given an (a, d_1, d_2, d_3) unlabeled BAST with $\left\lfloor \frac{ag}{2} \right\rfloor > d_1$, the maximum number of unsatisfied CNs, $b_{\text{b\_max}}$, in the resulting BAST after edge labeling is given by:

$$b_{\text{b\_max}} = d_1 + b_{b\_ut}, \text{ where} \tag{3.12}$$

$$b_{b\_ut} = \left\lfloor \frac{1}{2} a \left\lfloor \frac{\gamma - 1}{2} \right\rfloor \right\rfloor - d_1. \tag{3.13}$$

Proof. From the definition of the EAS, every unsatisfied CN in an EAS must be of degree 1. Thus, $b_{\text{e\_max}} = b = d_1$.

On the other hand, from the definition of the BAS (see Chapter 2), the maximum number of unsatisfied CNs it can have is (applies also to BASTs):

$$b_{\text{b\_max}} = \left\lfloor \frac{ag}{2} \right\rfloor. \tag{3.14}$$

However, we can also write $b_{\text{b\_max}}$ as:

$$b_{\text{b\_max}} = d_1 + b_{b\_ut}, \tag{3.15}$$

where b_{b_ut} is the maximum number of degree-2 unsatisfied CNs ($\in \mathcal{T}$) in the BAST. Combining (3.14) and (3.15), and recalling that $g = \left\lfloor \frac{\gamma-1}{2} \right\rfloor$ complete the proof. $\square$

Now, for a given $(a, d_1, d_1, d_2, 0)$ EAS ($b = d_1$ and $d_3 = 0$ for any EAS), let $\mathcal{Z}_{\text{e}}$ be the set of all EASs which have the same unlabeled EAS configuration as the original $(a, d_1, d_1, d_2, 0)$ EAS. The set $\mathcal{Z}_{\text{e}}$ contains only one element, which is the given EAS itself. Additionally, for a given (a, b, d_1, d_2, d_3) BAST, let $\mathcal{Z}_{\text{b}}$ be the set of all (a, b'_b, d_1, d_2, d_3) BASTs with $d_1 \leqslant b'_b \leqslant b_{\text{b\_max}}$ which have the same unlabeled BAST configuration as the original (a, b, d_1, d_2, d_3) BAST. Here, $b_{\text{b\_max}} = \left\lfloor \frac{ag}{2} \right\rfloor$ is the largest allowable number of unsatisfied CNs for these configurations. In the following lines, we rewrite Definitions 8 and 9 to make them suitable for EASs and BASTs.

Definition 10. *An $(a, d_1, d_1, d_2, 0)$ **EAS** (resp., (a, b, d_1, d_2, d_3) **BAST**) is said to be **removed** from the Tanner graph of an NB-LDPC code if and only if the resulting object (after edge weight processing) $\notin \mathcal{Z}_{\text{e}}$ (resp., $\notin \mathcal{Z}_{\text{b}}$).*

Definition 11. *For a given $(a, d_1, d_1, d_2, 0)$ EAS and its associated adjacency matrix $\mathbf{A}$ and its associated set $\mathcal{Z}_{\text{e}}$, we construct a set of 1 matrix as follows: $\mathbf{W}_1^{\text{e\_cm}}$ is an $(\ell - d_1) \times a$ submatrix, formed by removing d_1 rows corresponding to CNs $\in \mathcal{O}$ from the $\ell \times a$ matrix $\mathbf{A}$ of the EAS. We refer to the matrix in this set as an **elementary weight consistency matrix (EWCM)** and to this set itself as $\mathcal{W}_{\text{e}}$.*

For a given BAST, define a matrix $\mathbf{W}_{\text{b}}^{\text{z}}$ to be the matrix obtained by removing b'_{b}, $d_1 \leqslant b'_b \leqslant b_{\text{b\_max}} = \left\lfloor \frac{ag}{2} \right\rfloor$, rows corresponding to CNs $\in (\mathcal{O} \cup \mathcal{T})$ from the matrix $\mathbf{A}$. These b'_{b} CNs can simultaneously be unsatisfied under some edge labeling that produces a BAST which has the same unlabeled BAST as the given BAST. Let $\mathcal{U}_{\text{b}}$ be the set of all matrices $\mathbf{W}_{\text{b}}^{\text{z}}$. Each element $\in \mathcal{Z}_{\text{b}}$ has one or more matrices $\in \mathcal{U}_{\text{b}}$.

Definition 12. *For a given (a, b, d_1, d_2, d_3) BAST and its associated adjacency matrix $\mathbf{A}$ and its associated set $\mathcal{Z}_{\text{b}}$, we construct a set of t_{b} matrices as follows:*

1. *Each matrix $\mathbf{W}_h^{\text{b\_cm}}$, $1 \leqslant h \leqslant t_{\text{b}}$, in this set is an $(\ell - b_h^{\text{b\_cm}}) \times a$ submatrix, $d_1 \leqslant b_h^{\text{b\_cm}} \leqslant b_{\text{b\_max}} = \left\lfloor \frac{ag}{2} \right\rfloor$, formed by removing* ***different*** *$b_h^{\text{b\_cm}}$ rows from the $\ell \times a$ matrix $\mathbf{A}$ of the BAST. These $b_h^{\text{b\_cm}}$ rows to be removed correspond to CNs $\in (\mathcal{O} \cup \mathcal{T})$ that can simultaneously be unsatisfied under some edge labeling that produces a BAST which has the same unlabeled BAST as the given BAST.*

2. *Each matrix $\mathbf{W}_{\text{b}}^{\text{z}} \in \mathcal{U}_{\text{b}}$, for every element $\in \mathcal{Z}_{\text{b}}$, contains at least one element of the resultant set as its submatrix.*

3. *This resultant set has the* ***smallest cardinality****, which is t_{b}, among all the sets which satisfy Conditions 1 and 2 stated above.*

We refer to the matrices in this set as ***balanced weight consistency matrices (BWCMs)****, and to this set itself as $\mathcal{W}_{\text{b}}$.*

We note that $b_{\text{e\_max}} \leqslant b_{\text{b\_max}} \leqslant b_{\text{max}}$. Thus, $|\mathcal{Z}_e| = 1 \leqslant |\mathcal{Z}_b| \leqslant |\mathcal{Z}|$. This condition ensures the simplicity of removing EASs compared with BASTs, and BASTs compared with GASTs. Such simplicity is a result of the extra degrees of freedom we gain when we customize the WCM definition and the removal process to focus only on the objects of interest. For example, since $|\mathcal{Z}_e| = 1$, to remove an EAS, it suffices to make a single edge weight change for any edge connected to a degree-2 CN to convert the object into another one $\notin \mathcal{Z}_{\text{e}}$. Further illustrations are provided in the following example.

Example 9. *Consider configuration (a) in Fig. 3.2, which is a $(6, 2, 0, 2, 8, 0)$ GAS ($\gamma = 3$). We note that the configuration is also a $(6, 2, 2, 8, 0)$ GAST, and it is at the same time an EAS (because $b = d_1$ and $d_3 = 0$) and a BAST (because $b = 2 < \left\lfloor \frac{ag}{2} \right\rfloor = 3$). From Lemma 4 and Theorem 2 we compute $b_{\text{e\_max}} = d_1 = 2$, $b_{\text{b\_max}} = \left\lfloor \frac{ag}{2} \right\rfloor = 3$, and $b_{\text{max}} = d_1 + \left\lfloor \frac{1}{2}(ag - d_1) \right\rfloor = 4$. If the channel used is symmetric, the code design objective becomes the removal of EASs [10], which means converting this object into another object $\notin \mathcal{Z}_e = \{(6, 2, 2, 8, 0)\}$. If the channel used is the PR channel, the code design objective* *becomes the removal of BASTs (see Chapter 2 and [40]), which means converting this object into another object $\notin \mathcal{Z}_b = \{(6,2,2,8,0),(6,3,2,8,0)\}$. If the channel used is asymmetric (e.g., the NLM or the CHMM channels), the objective becomes the removal of GASTs, which means converting the object into another one $\notin \mathcal{Z} = \{(6,2,2,8,0),(6,3,2,8,0),(6,4,2,8,0)\}$.*

Note that once the WCMs are properly determined (EWCM or BWCMs), Theorem 3 is applied to remove any EAS or BAST.

3.3.4 Parent and Child GASTs

In graph theory, if Graph 1 is a subgraph of Graph 2, the former is called **parent** and the latter is called **child**. It has been shown in [10] that removing the parent EAS removes the child EAS. The analysis in the last subsection illustrates why this is the case (because only one edge weight change is needed to remove both the parent and the child EASs). However, this is not the case for GASTs because the GAST removal process is more complicated. We emphasize on this observation using the following lemma.

Lemma 5. *Consider an $(a'', b'', d_1'', d_2'', d_3'')$ GAST which is a parent of an (a, b, d_1, d_2, d_3) GAST. Removing the parent GAST does not guarantee removing the child GAST. The removal here is according to Definition 8.*

Proof. We prove this lemma by an example which shows that removing the parent GAST does not necessarily result in removing the child GAST. Consider the VN v which is shared between the parent GAST and the child GAST.

Suppose that v is connected to exactly $\left\lceil \frac{\gamma+1}{2} \right\rceil$ satisfied CNs in the parent GAST and strictly more than $\left\lceil \frac{\gamma+1}{2} \right\rceil$ satisfied CNs in the child GAST. Making one more CN connected to v unsatisfied (by properly changing one of the weights of the edges connected to it) changes the number of satisfied CNs connected to v to be $\left\lceil \frac{\gamma+1}{2} \right\rceil - 1$ in the parent GAST, and to be $\geqslant \left\lceil \frac{\gamma+1}{2} \right\rceil$ in the child GAST. Thus, the parent GAST, which is the $(a'', b'', d_1'', d_2'', d_3'')$ GAST, is removed (converted into a non-AS). On the contrary, the child GAST is now converted into another GAST, which is an $(a, b+1, d_1, d_2, d_3)$ GAST. $\square$

The following example serves to illustrate Lemma 5.

Example 10. *Recall the* $(6, 0, 0, 9, 0)$ *GAST* ($\gamma = 3$) *in Fig. 3.4 (b). We note that the VNs* $\{v_1, v_2, v_3, v_4\}$ *form a* $(4, 4, 4, 4, 0)$ *GAST that has* $\mathcal{T} = \{c_1, c_2, c_3, c_8\}$ *and* $\mathcal{O} = \{c_6, c_7, c_9, c_4\}$*. The* $(4, 4, 4, 4, 0)$ *GAST, which can only be an elementary AS* ($|\mathcal{Z}| = 1$)*, is the parent GAST, and the* $(6, 0, 0, 9, 0)$ *GAST is the child GAST. If we change the edge weight* w_{11}*, the parent GAST is completely removed as it has* $|\mathcal{Z}| = 1$ *(see also Remark 9). However, the child GAST is converted into a* $(6, 1, 0, 9, 0)$ *GAST, with one degree-*2 *unsatisfied CN.*

3.4 The WCM Optimization Framework

In this section, we deploy all the illustrated definitions and theorems to develop the new NB-LDPC code optimization framework. The objective of the new framework is to remove the detrimental objects (GASTs, EASs, or BASTs) from the Tanner graph of the NB-LDPC code using their WCMs and via edge weight manipulation.

3.4.1 Extracting the WCMs

First, we separately introduce Algorithm 2 for finding the WCMs. Algorithm 2 operates mainly on the unlabeled configuration of the object (the unlabeled GAST, unlabeled EAS, or unlabeled BAST) to determine the WCMs.

For the sake of clarity, we show the version of the algorithm which deals with GASTs. For EASs (resp., BASTs), $\mathcal{W}$ should be replaced by $\mathcal{W}_{\text{e}}$ (resp., $\mathcal{W}_{\text{b}}$) representing EWCMs (resp., BWCMs). Moreover, for EASs, there is only one EWCM of size $(\ell - d_1) \times a$. For BASTs, b_{ut} in Algorithm 2 should be replaced by b_{b_ut} given by (3.13), and t should be replaced by t_{b}, while the rest of the algorithm stays the same.

Algorithm 2 Finding the WCMs of a Given GAST

```
1: Input: Tanner graph $G_s$ of the GAST $s$, with edge weights over GF$(q)$, from which, the matrix $\mathbf{A}$ is formed.
2: Set the maximum number of nested for loops, loop_max.
3: Mark all the CNs $\in (\mathcal{T} \cup \mathcal{H})$ as satisfied. (CNs $\in \mathcal{O}$ are always unsatisfied.)
4: Check if $\exists$ in $G_s$ at least one degree-2 CN connecting two VNs, each is connected to $> \left\lceil \frac{\gamma+1}{2} \right\rceil$ CNs that are marked as satisfied.
5: if $\nexists$ any of them then
6:     $\exists$ only one $(\ell - d_1) \times a$ WCM. Extract it by removing all the rows corresponding to degree-1 CNs from the matrix $\mathbf{A}$.
7:     Go to 26.
8:     else
9:     Count such CNs (that satisfy the condition in 4), save the number in $u^0$, and save their indices (the indices of their rows in $\mathbf{A}$) in $\mathbf{y}^0 = [y^0(1)\ y^0(2)\ \dots\ y^0(u^0)]^{\mathrm{T}}$.
10: end if
11: Compute $b_{\mathrm{ut}}$ from (3.5) in Theorem 2. If $b_{\mathrm{ut}} = 1$, go to 25.
12: for $i_1 \in \{1, 2, \dots, u^0\}$ do (Level 1)
13:     Remove the marking performed in levels $\geqslant 1$, and mark the selected CN $c_{y^0(i_1)}$ as unsatisfied.
14:     Redo the counting in 9, but save in $u^1_{i_1}$ $(< u^0)$ and $\mathbf{y}^1_{i_1}$ (instead of $u^0$ and $\mathbf{y}^0$, resp.).
15:     If $b_{\mathrm{ut}} = 2 \parallel u^1_{i_1} = 0$, go to 12.
16:     for $i_2 \in \{1, 2, \dots, u^1_{i_1}\}$ do (Level 2)
17:         Remove the marking performed in levels $\geqslant 2$, and mark the selected CN $c_{y^1_{i_1}(i_2)}$ as unsatisfied.
18:         Redo the counting in 9, but save in $u^2_{i_1,i_2}$ $(< u^1_{i_1})$ and $\mathbf{y}^2_{i_1,i_2}$.
19:         If $b_{\mathrm{ut}} = 3 \parallel u^2_{i_1,i_2} = 0$, go to 16.
20:         ...
21:         The lines from 16 to 19 are repeated (loop_max$-2$) times, with the nested (loop_max$-2$) for loops executed over the running indices $i_3$, $i_4$, ..., $i_{\mathrm{loop\_max}}$.
22:         ...
23:     end for
24: end for
25: Obtain the WCMs via the indices in the y arrays. In particular, by removing permutations of the rows corresponding to $c_{y^0(i_1)}, c_{y^1_{i_1}(i_2)}, \dots, c_{y^{b_{\mathrm{ut}}-1}_{i_1,i_2,\dots,i_{b_{\mathrm{ut}}-1}}(i_{b_{\mathrm{ut}}})}$, and the degree-1 CNs from $\mathbf{A}$, we can reach any WCM.
26: Eliminate all the repeated WCMs to reach the final set of WCMs, $\mathcal{W}$, where $|\mathcal{W}| = t$.
27: Output: The set $\mathcal{W}$ of all WCMs of the GAST.
```

Note that for GASTs, b_{ut} is an upper bound for the maximum number of rows corresponding to degree-2 CNs that can be removed from $\mathbf{A}$. Thus, it can happen in some cases that $u^{b_{\mathrm{ut}}-1}_{i_1,i_2,\dots,i_{b_{\mathrm{ut}}-1}} = 0$, $\forall i_1, i_2, \dots, i_{b_{\mathrm{ut}}-1}$. In such cases, the exact maximum number of rows corresponding to degree-2 CNs that can be removed from $\mathbf{A}$ is the number of levels (nested loops in Algorithm 2), denoted by b_{et}, after which $u^{b_{\text{et}}}_{i_1,i_2,\dots,i_{b_{\text{et}}}} = 0$, $\forall i_1, i_2, \dots, i_{b_{\text{et}}}$.

Note also that because the WCMs do not necessarily have the same row dimension, Algorithm 2 may stop before reaching b_{et} levels starting from some $c_{y^0(i_1)}$, which results in an $(\ell - b^{\text{cm}}_h) \times a$ WCM with $b^{\text{cm}}_h < b_{\max} = d_1 + b_{\text{et}}$.

Example 11. *To illustrate Algorithm 2, we contrast the two configurations in Fig. 3.4 one more time. The set $\mathcal{W}$ contains only one WCM of size 6×4 for the $(4,4,4,6,0)$ GAST ($\gamma = 4$) (see Steps 4, 5, 6, and 7 of Algorithm 2). Having a single WCM is the case for all GASTs that cannot have unsatisfied CNs $\in \mathcal{T}$, which exemplifies the ease in removing such GASTs. On the contrary, for the $(6,0,0,9,0)$ GAST ($\gamma = 3$), following Algorithm 2 gives $u^0 = 9$ (i.e., all the CNs are connecting pairs satisfying the condition in Theorem 1), $\mathbf{y}^0 = [1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9]^{\text{T}}$, and $b_{\max} = b_{\text{ut}} = b_{\text{et}} = 3$. The matrix $\mathbf{A}$ is:*

$$\mathbf{A} = \begin{array}{c} \\ c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \\ c_8 \\ c_9 \end{array} \begin{array}{c} \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array} \\ \begin{bmatrix} w_{1,1} & w_{1,2} & 0 & 0 & 0 & 0 \\ 0 & w_{2,2} & w_{2,3} & 0 & 0 & 0 \\ 0 & 0 & w_{3,3} & w_{3,4} & 0 & 0 \\ 0 & 0 & 0 & w_{4,4} & w_{4,5} & 0 \\ 0 & 0 & 0 & 0 & w_{5,5} & w_{5,6} \\ w_{6,1} & 0 & 0 & 0 & 0 & w_{6,6} \\ 0 & w_{7,2} & 0 & 0 & w_{7,5} & 0 \\ w_{8,1} & 0 & 0 & w_{8,4} & 0 & 0 \\ 0 & 0 & w_{9,3} & 0 & 0 & w_{9,6} \end{bmatrix} \end{array}.$$

*If Algorithm 2 selects c_1 first, c_1 will be marked as unsatisfied. As a result, c_2, c_7, c_6, and c_8 cannot be selected with c_1 (otherwise the configuration will not be an AS). On level 2 (the second loop in the group of nested **for** loops in Algorithm 2), the algorithm can select one of c_3, c_4, c_5, and c_9. If the algorithm selects c_3, c_3 will be marked as unsatisfied. As a result, c_4 and c_9 cannot be selected. Thus, the only choice remaining on level 3 (the last level) is c_5, which means one WCM is extracted by removing the rows corresponding to (c_1, c_3, c_5) together* *from* $\mathbf{A}$*. Applying the steps from 12 to 26 on all the* 9 *CNs in* $\mathbf{y}^0$*, results in that the set* $\mathcal{W}$ *contains* 6 *WCMs of size* 6×6*. They are formed by removing the rows corresponding to the following groups of* 3 *CNs from* $\mathbf{A}$*:* $\{(c_1, c_3, c_5)$, (c_1, c_4, c_9), (c_2, c_4, c_6), (c_2, c_5, c_8), (c_3, c_6, c_7), $(c_7, c_8, c_9)\}$.

3.4.2 The New NB-LDPC Code Optimization Algorithm

We are now ready to present our optimization algorithm. Again, for the sake of clarity, we show the version of the algorithm which deals with GASTs. For EASs (resp., BASTs), unlabeled GAST and WCM should be changed into unlabeled EAS and EWCM (resp., unlabeled BAST and BWCM). Additionally, for EASs (resp., BASTs), $E_{\text{GAST,min}}$ and $\mathbf{W}_h^{\text{cm}}$ should be changed into $E_{\text{EAS,min}}$ (which equals 1) and $\mathbf{W}_h^{\text{e\_cm}}$ (resp., $E_{\text{BAST,min}}$ and $\mathbf{W}_h^{\text{b\_cm}}$). Obviously, many steps in Algorithm 3 will be skipped in case the objective is to remove EASs because in this case, $|\mathcal{W}_e| = 1$ and a single edge weight change (for any edge connected to a degree-2 CN) is sufficient to remove the target object. For BASTs, t should be changed into t_b, where $|\mathcal{W}_b| = t_b$.

Remark 12. *For EASs, resolving the null space of the single EWCM according to Algorithm 3 is the same as breaking the weight condition stated in [10, Lemma 1].*

Remark 13. *The process of determining the set* $\mathcal{G}$ *of GASTs to be removed (Step 2 in Algorithm 3) is summarized as follows. First, we identify the dominant GASTs (which are the GASTs that contribute the most to the error profile in the error floor region) by initial simulations (see also [23]) in addition to combinatorial properties of the code. Second, we determine the set* $\mathcal{G}$ *using two complementary methods. The first method is to use efficient combinatorial techniques (e.g., [6]) to locate the* (a, d_1, d_2, d_3) *unlabeled GASTs, for a given dominant* (a, b, d_1, d_2, d_3) *GAST, in the Tanner graph of the code. The second method is to use additional extensive simulations for the non-binary code to locate GASTs, and for the binary code, which corresponds to the unlabeled Tanner graph, to locate unlabeled GASTs. For*

Algorithm 3 Optimizing NB-LDPC Codes by Reducing the Number of GASTs

```
 1: Input: Tanner graph $G$ of the NB-LDPC code with edge weights over GF($q$).
 2: Using initial simulations and combinatorial techniques (e.g., [6]), determine $\mathcal{G}$, the set of
    GASTs to be removed.
 3: Let $\mathcal{X}$ be the set of GASTs in $\mathcal{G}$ that cannot be removed, and initialize it with $\varnothing$.
 4: Let $\mathcal{P}$ be the set of GASTs in $\mathcal{G}$ that have been processed, and initialize it with $\varnothing$.
 5: Sort the GASTs in $\mathcal{G}$ according to their sizes (parameter $a$) from the smallest to the
    largest.
 6: Start from the smallest GAST (smallest index).
 7: for every GAST $s \in \mathcal{G} \setminus \mathcal{P}$ do
 8:    If the unlabeled configuration of $s$ does not satisfy the unlabeled GAST conditions in
    Definitions 6 and 7, skip $s$ and go to 7.
 9:    Determine the minimum number of necessary edge weight changes to remove the
    GAST $s$, $E_{\text{GAST,min}}$, by using Lemma 2 in Chapter 2.
10:    Extract the subgraph $G_s$ of the GAST $s$, from $G$.
11:    Use Algorithm 2 to determine the set $\mathcal{W}$ of all WCMs of $s$ ($|\mathcal{W}| = t$).
12:    for $h \in \{1, 2, \dots, t\}$ do
13:       Find the null space $\mathcal{N}(\mathbf{W}_h^{\text{cm}})$ of the $h$th WCM.
14:       if (3.8) is satisfied (i.e., the WCM already has broken weight conditions) then
15:          Go to 12.
16:       else
17:          Keep track of the changes already performed in $G_s$. (The total number of changes
    to remove the GAST should be as close as possible to $E_{\text{GAST,min}}$.)
18:          Determine the smallest set of edge weight changes in $G_s$ needed to achieve (3.8)
    for the $h$th WCM, without violating (3.8) for WCMs prior to the $h$th.
19:          If this set of edge weight changes does not undo the removal of any GAST $\in \mathcal{P} \setminus \mathcal{X}$,
    perform these changes in $G_s$ and go to 12.
20:          if $\nexists$ more edge weights to execute 18 and 19 then
21:             Add GAST $s$ to the set $\mathcal{X}$ and go to 27.
22:          else Go to 18 to determine a new set of changes.
23:          end if.
24:       end if
25:    end for
26:    Update $G$ by the changes performed in $G_s$.
27:    Add GAST $s$ to the set $\mathcal{P}$.
28:    If $\mathcal{P} \neq \mathcal{G}$, go to 7 to pick the next smallest GAST.
29: end for
30: If $\mathcal{X} = \varnothing$, then all the GASTs have been removed. Otherwise, only the remaining GASTs
    in $\mathcal{X}$ cannot be removed.
31: Output: Updated Tanner graph $G$ of the optimized NB-LDPC code with edge weights
    over GF($q$).
```

both methods, given the unlabeled GAST, the WCM framework is used to detect whether the edge weights are such that the labeled configuration is indeed a GAST (see also Remark 11).

The removal of a specific GAST might result in undoing the removal of another GAST if the removal of the former is done via changing the weight (weights) of an edge (edges) shared between the two GASTs. That is why Step 19 in Algorithm 3 is needed.

Note that the complexity of the process of removing a specific GAST using the WCM framework is mainly controlled by the number of WCMs, which is t, of that GAST (see the **for** loop in Step 12 of Algorithm 3). Thus, the complexity of Algorithm 3 depends on the size of the set $\mathcal{G}$ and the numbers of WCMs of the GASTs in $\mathcal{G}$.

As long as the maximum degree of any unsatisfied CN in the AS (or GAS) is 2, Algorithm 3 can be used to remove any type of ASs from the NB-LDPC code. Algorithm 3 can be used to remove EASs, BASTs, and of course GASTs. That is the reason why the proposed WCM optimization framework is general, in the sense that it is suitable for optimizing NB-LDPC codes to be used over many channels (e.g., the AWGN channel, the PR channel, the CHMM channel, and even an aggressively asymmetric channel like the NLM channel to be discussed in the next section).

3.5 Applications in Practical Channels

In this section, we demonstrate the effectiveness of the WCM framework by simulating different NB-LDPC codes optimized for various channels and applications. We used a finite-precision, fast Fourier transform based q-ary sum-product algorithm (FFT-QSPA) LDPC decoder [52] to generate all the results. The decoder performs a maximum of 50 iterations (except for the PR channel simulations), and it stops if a codeword is reached sooner.

All the unoptimized NB-LDPC codes we are using are regular protograph-based NB-PB-LDPC codes. These codes are constructed as follows. First, a binary protograph matrix $\mathbf{H}^{\text{p}}$ is designed. Then, $\mathbf{H}^{\text{p}}$ is lifted via a lifting parameter ζ to create the binary image of $\mathbf{H}$, which is $\mathbf{H}^{\mathrm{b}}$. The lifting process means that every 1 in $\mathbf{H}^{\mathrm{p}}$ is replaced by a $\zeta \times \zeta$ circulant matrix, while every 0 (if any) in $\mathbf{H}^{\mathrm{p}}$ is replaced by a $\zeta \times \zeta$ all-zero matrix. The circulant powers are adjusted such that the unlabeled Tanner graph of the resulting code does not have cycles of length 4. Then, the 1's in $\mathbf{H}^{\mathrm{b}}$ are replaced by non-zero values $\in \mathrm{GF}(q)$ to generate $\mathbf{H}$. These unoptimized codes are high performance NB-PB-LDPC codes (see also [8] and [9]). Note that while we are focusing on PB structured codes in our simulations, the WCM framework works for any regular NB-LDPC codes. Moreover, the WCM framework also works for any GF size q and for any code rate.

Remark 14. *While we focus in this work on regular NB-LDPC codes with fixed column and row weights, the WCM framework can also be applied to NB-LDPC codes that have only the column weight fixed (i.e., fixed VN degree).*

Here, RBER is the raw bit error rate. If we define the data read out of the Flash memory without error correction as the raw data, then RBER equals the number of raw data bits in error divided by the total number of raw data bits read [49]. Furthermore, UBER is the uncorrectable bit error rate, which is a metric for the fraction of bits in error out of all bits read after the error correction is applied [49]. A useful formulation of UBER, which is recommended by industry, is the frame error rate (FER) divided by the sector size in bits.

3.5.1 Results for Practical Flash Channels

In this subsection, we present our results over two practical Flash channels, namely the NLM [24] and the CHMM [25] channels. While the two channels are asymmetric, the NLM channel incorporates more asymmetry because of the way it models programming errors.

We start first with the NLM channel; the authors in [24] accurately modeled the threshold voltage distribution of sub-20nm 4-level (2-bit, multi-level cell (MLC)) Flash memories. The four levels (states) are modeled as different normal-Laplace mixture distributions, taking into account various sources of error due to wear-out effects, e.g., programming errors (significant asymmetry). Through device testing, the authors provided accurate fitting results of their model for program/erase (P/E) cycles up to 10 times the manufacturer's endurance specification. We implemented the NLM channel based on the parameter set described in [24]. We use 3 reads, and the sector size is 512 bytes.

In the NLM simulations, Code 3.1, that was introduced in Example 6, is again an NB-PB-LDPC code defined over GF(4), with block length = 3,996 bits, rate ≈ 0.89, and $\gamma = 3$. Code 3.2 is an NB-PB-LDPC code defined over GF(4), with block length = 3,280 bits, rate ≈ 0.80, and $\gamma = 4$. Code 3.3 (resp., Code 3.4) is the result of optimizing Code 3.1 (resp., Code 3.2) for symmetric channels by attempting to remove only the EASs in Table 3.1 (resp., Table 3.2). Note that EASs have $b = d_1$ and $d_3 = 0$. Code 3.5 (resp., Code 3.6) is the result of optimizing Code 3.1 (resp., Code 3.2) for the NLM channel (which is asymmetric) by attempting to remove the GASTs in Table 3.1 (resp., Table 3.2) using the WCM framework.

Figures 3.5 and 3.6 show that even though only 3 reads are used, the codes optimized by removing GASTs using the WCM framework (Codes 3.5 and 3.6) outperform the unoptimized codes (Codes 3.1 and 3.2) by more than 1 order of magnitude. More intriguingly, the two figures show that conventional code optimization techniques which assume channel symmetry (e.g., techniques that would focus on the removal of EASs) are ineffective for realistic memory (storage) channels (Codes 3.5 and 3.6 outperform Codes 3.3 and 3.4, respectively, by over 0.5 of an order of magnitude).

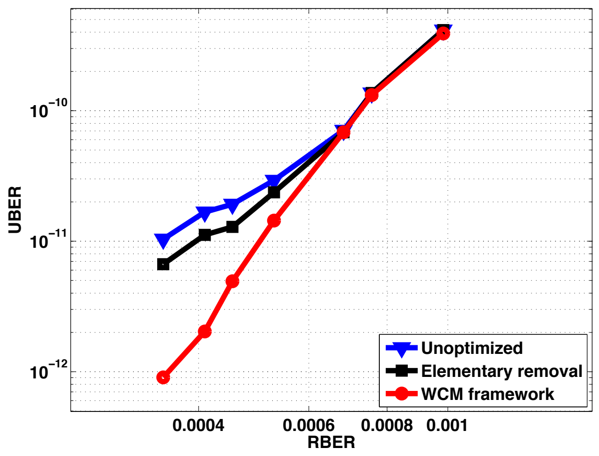


Figure 3.5: Simulation results over the NLM channel for Code 3.1 (unoptimized), Code 3.3 (elementary removal), and Code 3.5 (WCM framework). All the three codes have $\gamma = 3$.

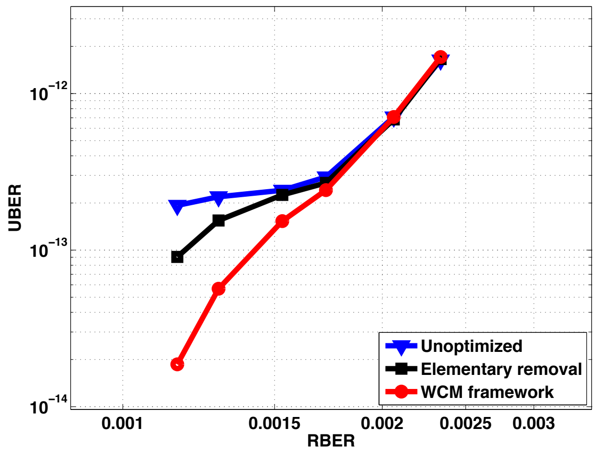


Figure 3.6: Simulation results over the NLM channel for Code 3.2 (unoptimized), Code 3.4 (elementary removal), and Code 3.6 (WCM framework). All the three codes have $\gamma = 4$.

Table 3.1: Error profile of Codes 3.1, 3.3, and 3.5 over the NLM channel, RBER $\approx 3.51 \times 10^{-4}$, UBER (unoptimized) $\approx 1.04 \times 10^{-11}$, and UBER (WCM framework) $\approx 9.04 \times 10^{-13}$ (see Fig. 3.5).

Error type	Count		
	Code 3.1	Code 3.3	Code 3.5
$(4, 2, 2, 5, 0)$	45	0	0
$(4, 3, 2, 5, 0)$	15	23	0
$(4, 4, 4, 4, 0)$	3	0	0
$(6, 0, 0, 9, 0)$	12	1	0
$(6, 1, 0, 9, 0)$	7	13	0
$(6, 2, 0, 9, 0)$	3	5	1
$(6, 2, 2, 5, 2)$	2	2	0
$(7, 1, 0, 10, 1)$	4	4	0
Other	9	13	8

Table 3.1 (resp., Table 3.2) shows the error profiles of Codes 3.1, 3.3, and 3.5 (resp., 3.2, 3.4, and 3.6). The tables reveal the effectiveness of the WCM framework in removing detrimental GASTs. The tables further illuminate why optimizing the codes by removing only EASs (as one would do for the case of symmetric channels) will not work for the NLM channel. First, unoptimized codes naturally have a high percentage of non-elementary ASs ($b > d_1$ or/and $d_3 > 0$) in their error profiles over the NLM channel (31% for Code 3.1, and 26% for Code 3.2). Second, any technique that primarily focuses on the elimination of EASs will not be effective as it would convert most of the elementary ASs into non-elementary ASs, which themselves are still problematic GASTs. For example, Code 3.3 has

Table 3.2: Error profile of Codes 3.2, 3.4, and 3.6 over the NLM channel, RBER $\approx 1.16 \times 10^{-3}$, UBER (unoptimized) $\approx 1.93 \times 10^{-13}$, and UBER (WCM framework) $\approx 1.86 \times 10^{-14}$ (see Fig. 3.6).

Error type	Count		
	Code 3.2	Code 3.4	Code 3.6
$(4,4,4,6,0)$	47	2	0
$(6,2,2,11,0)$	11	0	0
$(6,4,2,11,0)$	14	14	0
$(6,4,4,7,2)$	6	6	0
$(8,3,2,15,0)$	6	6	1
$(8,4,4,14,0)$	5	1	0
Other	11	18	9

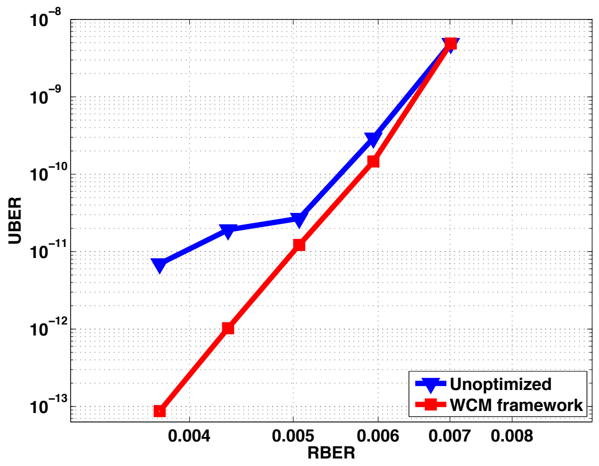


Figure 3.7: Simulation results over the NLM channel for Code 3.7 (unoptimized) and Code 3.8 (WCM framework). The two codes have $\gamma = 4$.

more $(4,3,2,5,0)$ GASTs in its error profile compared with Code 3.1 (see Table 3.1) mainly because many of these $(4,3,2,5,0)$ GASTs were originally $(4,2,2,5,0)$ GASTs (elementary), and the optimization procedure of removing EASs has converted them into $(4,3,2,5,0)$ GASTs (non-elementary).

Furthermore, we introduce results for Code 3.7, which is an NB-PB-LDPC code defined over GF(4), with block length = 8,480 bits, rate ≈ 0.90, and $\gamma = 4$, in addition to Code 3.8, which is the result of optimizing Code 3.7 for the NLM channel by attempting to remove the dominant GASTs $(4,4,4,6,0)$, $(6,4,4,10,0)$, $(6,5,5,8,1)$, and $(8,4,2,15,0)$. Fig. 3.7 demonstrates that a performance gain of nearly 2 orders of magnitude is achievable via optimizing practical codes like Code 3.7 using the WCM framework to reach Code 3.8.

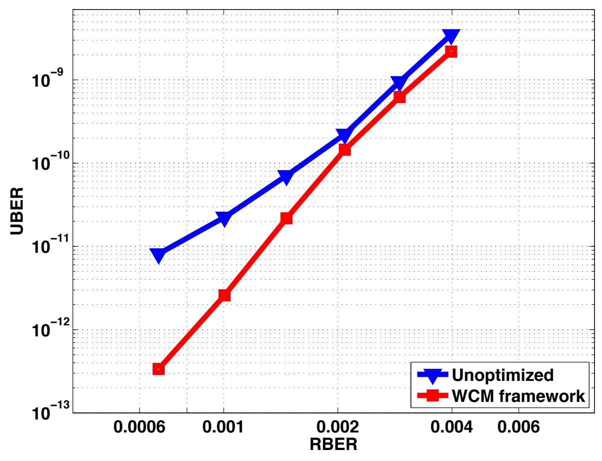


Figure 3.8: Simulation results over the CHMM channel for Code 3.1 (unoptimized) and Code 3.10 (WCM framework). The two codes have $\gamma = 3$.

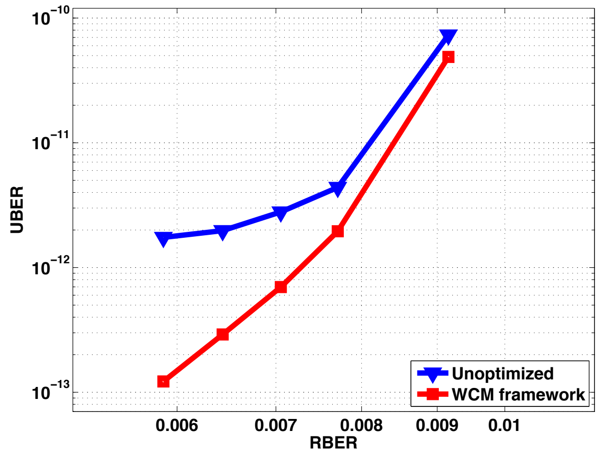


Figure 3.9: Simulation results over the CHMM channel for Code 3.9 (unoptimized) and Code 3.11 (WCM framework). The two codes have $\gamma = 4$.

We also provide results over another Flash channel, which is the CHMM channel; the authors in [25] developed a model for the threshold voltage distribution that is suitable for 20nm and 24nm 4-level (MLC) Flash memories. They modeled the four levels (states) as different Gaussian distributions, along with additive white noise, that are shifted and broadened as P/E cycles increase (limited asymmetry). We implemented the CHMM channel using the data and the model provided in [25]. We use 3 reads, and the sector size is 512 bytes.

In the CHMM simulations, we reuse Code 3.1 (that has $\gamma = 3$). Code 3.9 is an NB-PB-

Table 3.3: Error profile of Codes 3.1 and 3.10 over the CHMM channel, RBER $\approx 6.74\times10^{-4}$, UBER (unoptimized) $\approx 8.13 \times 10^{-12}$, and UBER (WCM framework) $\approx 3.37 \times 10^{-13}$ (see Fig. 3.8).

Error type	Count	
	Code 3.1	Code 3.10
$(4, 2, 2, 5, 0)$	52	0
$(5, 2, 2, 5, 1)$	3	0
$(5, 3, 3, 6, 0)$	7	0
$(6, 0, 0, 9, 0)$	4	0
$(6, 1, 1, 7, 1)$	8	0
$(6, 2, 0, 9, 0)$	12	0
$(6, 2, 2, 8, 0)$	3	0
$(7, 1, 0, 10, 1)$	2	1
$(8, 0, 0, 12, 0)$	2	0
Other	7	4

Table 3.4: Error profile of Codes 3.9 and 3.11 over the CHMM channel, RBER $\approx 5.87\times10^{-3}$, UBER (unoptimized) $\approx 1.74 \times 10^{-12}$, and UBER (WCM framework) $\approx 1.22 \times 10^{-13}$ (see Fig. 3.9).

Error type	Count	
	Code 3.9	Code 3.11
$(4, 4, 4, 6, 0)$	38	0
$(6, 4, 2, 11, 0)$	2	0
$(6, 4, 4, 10, 0)$	33	0
$(7, 4, 3, 11, 1)$	12	0
$(8, 5, 5, 12, 1)$	5	0
$(9, 5, 5, 14, 1)$	2	1
Other	8	6

LDPC code defined over GF(4), with block length = 1,840 bits, rate $\approx$ 0.80, and $\gamma = 4$. Code 3.10 (resp., Code 3.11) is the result of optimizing Code 3.1 (resp., Code 3.9) for the CHMM channel (which is asymmetric) by attempting to remove the GASTs in Table 3.3 (resp., Table 3.4) using the WCM framework.

Figures 3.8 and 3.9 confirm, on a different Flash channel, that even though only 3 reads are used, the codes optimized using the WCM framework (Codes 3.10 and 3.11) outperform the unoptimized codes (Codes 3.1 and 3.9) by more than 1 order of magnitude (nearly 1.4 orders in Fig. 3.8).

Table 3.3 (resp., Table 3.4) shows the error profiles of Codes 3.1 and 3.10 (resp., 3.9 and 3.11). The tables again reveal the effectiveness of the WCM framework in removing detrimental GASTs. An interesting observation from Tables 3.1, 3.2, 3.3, and 3.4 is that the percentage of non-elementary ASs ($b > d_1$ or/and $d_3 > 0$) in the error profile of a code is a function of the asymmetry the channel incorporates. For example, Table 3.1 shows that the percentage of non-elementary ASs in the error profile of Code 3.1 over the NLM channel is 31%. However, this percentage drops to 25% for the same code over the CHMM channel, as Table 3.3 reveals. The reason is that the NLM channel incorporates more asymmetry.

3.5.2 Results for Other Channels

In this subsection, we present additional results on channels that are not Flash-related. In particular, we present simulation results over the PR channel (encountered in 1-D MR applications) and the AWGN channel (as an example on canonical symmetric channels).

The PR channel used is the one described in Chapter 2. The channel incorporates inter-symbol interference (intrinsic memory) along with jitter and electronic noise. The normalized channel density [51] is 1.4 while the PR equalization target is [8 14 2]. The number of global (detector-decoder) iterations is either 5 or 10 (see Fig. 3.10), and the maximum number of local (decoder only) iterations is 20. More details can be found in Chapter 2.

In the PR simulations, Code 3.12 is an NB-PB-LDPC code defined over GF(8), with block length = 867 bits, rate $\approx$ 0.82, and $\gamma = 3$. Code 3.13 is the result of optimizing Code 3.12 by attempting to remove the dominant BASTs $(6, 0, 0, 9, 0)$, $(6, 1, 0, 9, 0)$, $(6, 2, 0, 9, 0)$, $(8, 0, 0, 12, 0)$, and $(10, 0, 0, 15, 0)$ using the WCM framework.

Fig. 3.10 shows that the code optimized using the WCM framework (Code 3.13) outperforms the unoptimized code (Code 3.12) by more than 1.5 orders of magnitude. Additionally, the optimized code (Code 3.13) outperforms the unoptimized code (Code 3.12) at half the latency by about 0.8 of an order of magnitude.

In the AWGN simulations, we reuse Code 3.9 (that has $\gamma = 4$). Code 3.14 is the result of optimizing Code 3.9 by attempting to remove the dominant EASs $(4, 4, 4, 6, 0)$, $(5, 2, 2, 9, 0)$, $(6, 2, 2, 11, 0)$, $(6, 4, 4, 10, 0)$, $(7, 4, 4, 12, 0)$, and $(8, 4, 4, 14, 0)$ using the WCM framework.

Fig. 3.11 shows that the code optimized using the WCM framework (Code 3.14) outperforms the unoptimized code (Code 3.9) by about 1.2 orders of magnitude.

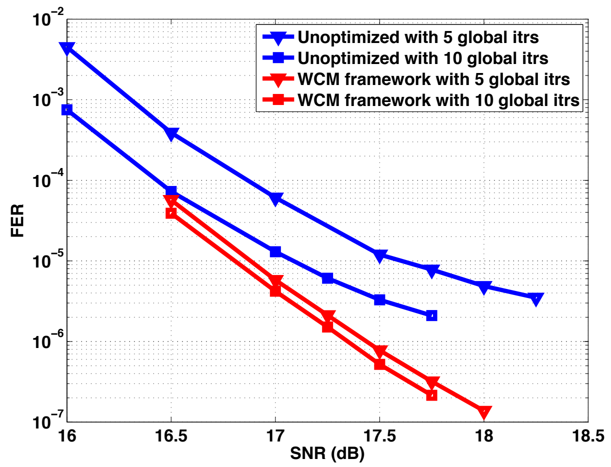


Figure 3.10: Simulation results over the PR channel for Code 3.12 (unoptimized) and Code 3.13 (WCM framework). The two codes have $\gamma = 3$, and they are defined over GF(8).

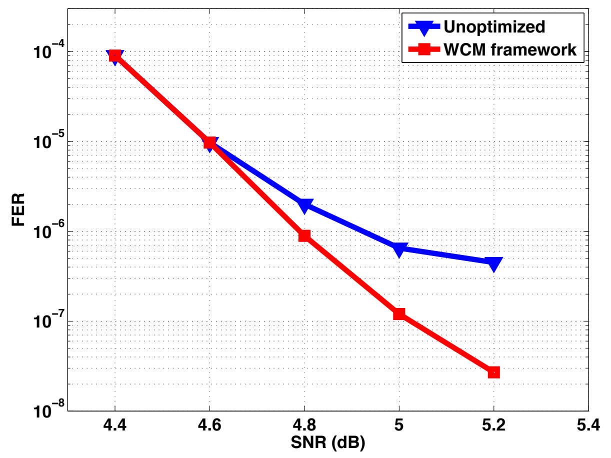


Figure 3.11: Simulation results over the AWGN channel for Code 3.9 (unoptimized) and Code 3.14 (WCM framework). The two codes have $\gamma = 4$.

In summary, the additional results in this subsection demonstrate that the WCM framework can effectively remove subclasses of GASTs, like BASTs and EASs, with the proper customization described in previous sections. The performance gains we achieve using the WCM framework over PR and AWGN channels are the same gains achieved in [40] and [10], respectively. Moreover, these results show that the WCM framework works for any GF size (see Fig. 3.10).

3.6 Concluding Remarks

We introduced a novel NB-LDPC code optimization framework based on new combinatorial definitions and linear-algebraic tools. The WCM framework was applied to codes used over realistic Flash channels, namely the NLM channel and the CHMM channel, where clear benefits of the proposed technique relative to the existing approaches were demonstrated. Moreover, our framework was shown to be effective in designing optimized NB-LDPC codes for MR applications (the PR channel) and for symmetric channels (the AWGN channel). As many emerging storage devices exhibit an increasing level of asymmetry (e.g., 3-D Flash), the presented framework can be a valuable code design and optimization tool that will enable data storage engineers to use LDPC codes with confidence.

Acknowledgement

The majority of the material in this chapter was published in [42]. The work was also presented in part at ISIT 2016 [41]. The author would like to thank the collaborators in these publications. Special thanks to Chinmayi Lanka for her contributions to the work.

CHAPTER 4

Analysis and Extensions of the WCM Framework

4.1 Introduction

Modern dense storage devices, e.g., multi-level Flash and magnetic recording (MR) devices, operate at very low frame error rate (FER) values, motivating the need for strong error correction techniques. Because of their capacity approaching performance, low-density parity-check (LDPC) codes [1, 59, 60, 61] are becoming the first choice for many of the modern storage systems [26, 27, 28, 51, 62, 63, 64]. Under iterative quantized decoding, LDPC codes suffer from the error floor problem, which is a change in the FER slope that undermines the chances of reaching desirable very low FER levels [14, 20, 21, 23, 65]. It was demonstrated in the literature that absorbing sets (ASs), which are detrimental subgraphs in the Tanner graph of the LDPC code, are the principal cause of the error floor problem [3, 10]. There are other works that studied different classes of detrimental objects, specifically, stopping sets [7] and trapping sets [5], [66]. Research works investigating the error floor problem of LDPC codes include [3, 4, 5, 14, 58, 66, 67, 68, 69, 70, 71].

Particularly for non-binary LDPC (NB-LDPC) codes, the authors in [10] used concepts from [7] to study non-binary elementary absorbing sets (EASs), and showed that EASs are the detrimental objects which contribute the most to the error floor of NB-LDPC codes over the canonical additive white Gaussian noise (AWGN) channel. The observation that the combinatorial structure of the dominant detrimental objects critically depends on the characteristics of the channel of interest was discussed first in Chapter 2 (see also [39] and [40]); we introduced balanced absorbing sets (BASs) and demonstrated their dominance in the error floor of NB-LDPC codes over partial-response (PR) channels, which exemplify 1-D MR channels [11, 12]. Motivated by the asymmetry possessed by practical Flash channels [24, 25], in Chapter 3, we introduced general absorbing sets (GASs) and general absorbing sets of type two (GASTs) to capture the dominant problematic objects over realistic Flash channels (see also [41] and [42]). GASs and GASTs subsume previously introduced AS subclasses (namely EASs, BASs, and BASTs).

In [10] and [40], NB-LDPC code optimization algorithms tailored to AWGN and PR channels, respectively, were proposed. While the weight consistency matrix (WCM) framework introduced in Chapter 3 was originally motivated by the need to optimize NB-LDPC codes for asymmetric Flash channels [41], we customized this methodology to be suitable for channels with memory (e.g., PR channels), canonical symmetric channels (e.g., AWGN channels), as well as practical Flash channels, achieving at least 1 order of magnitude performance gain over all these channels. As illustrated in Chapter 3, the principal idea of the WCM framework is representing a problematic object, e.g., a GAST, using a small set of matrices, called WCMs. Since problematic objects in an NB-LDPC code are described in terms of both their weight conditions as well as their topological conditions, there are explicit weight properties associated with the WCMs of an object. By changing the null spaces of the WCMs associated with an object such that the weight conditions of all these WCMs are broken, this problematic object is removed from the Tanner graph of the code. A key feature of the WCM framework is that the GASTs removal process is performed solely via manipulating the edge weights of the Tanner graph of the NB-LDPC code, which consequently preserves all the structural topological properties of the code being optimized.

For NB-LDPC codes with fixed column weights (variable node degrees), our contributions in this chapter are:

1. We characterize GASTs via their WCMs. In particular, we define the unlabeled GAST

tree to describe the underlying topology of a GAST, where the leaves of this tree represent the WCMs of the GAST. Using this tree, we prove the optimality of the WCM framework by demonstrating that the framework indeed operates on the minimum possible number of matrices to remove the detrimental object. We also deploy concepts from graph theory and combinatorics to compute the exact number of WCMs associated with a GAST in different cases. We further compare the number of matrices the WCM framework operates on with the number of matrices a suboptimal idea works with, showing the significant reduction (up to about 90%) achieved by the WCM framework in the cases of interest.

2. Based on tools from graph theory and linear algebra, we propose a comprehensive analysis of the removal process of GASTs. We start off with discussing the dimensions of the null spaces of WCMs; these null spaces play the central role in the identification and removal of a GAST. Then, we derive the best that can be done to process a short WCM (a WCM that has fewer rows than columns) during the GAST removal process. Finally, we provide the minimum number of edge weight changes[1] needed to remove a GAST, along with how to select the edges and the new weights to guarantee the removal of the GAST through its WCMs.

3. We introduce new combinatorial objects that capture the majority of the non-GAST detrimental objects in the error floor region of NB-LDPC codes that have even column weights over asymmetric Flash channels. We define oscillating sets (OSs) and oscillating sets of type two (OSTs). Furthermore, we expand the analysis of GASTs in Chapter 3 to cover OSTs, describing how the WCM framework can be customized to remove OSTs, after GASTs have been removed, from the Tanner graph of a code to achieve additional performance gains.

[1]In the WCM framework, a GAST is removed via careful processing of the weights of its edges (the original and the new weights are not zeros). Throughout this chapter, the edge weight changes are always with respect to the original configuration.

4. We extend the scope of the WCM framework by using it to optimize codes with different properties and for various applications. Specifically, we show that despite the good error floor performance of NB-LDPC codes with column weight 5 before optimization, more than 1 order of magnitude gain in the uncorrectable bit error rate (UBER) over practical Flash channels is achievable via the WCM framework. We further apply the theoretical concepts in Item 3 for NB-LDPC codes with column weight 4 over practical Flash channels to achieve overall UBER gains up to nearly 2.5 orders of magnitude. Additionally, we optimize NB-LDPC codes for practical Flash channels with more soft information (6 reads). We also use the WCM framework to optimize NB-LDPC codes with irregular check node (CN) degrees and fixed variable node (VN) degrees; we show that more than 1 order of magnitude performance gain in the FER is achievable by optimizing spatially-coupled (SC) codes [30, 31, 33, 72, 73, 74] used over PR and AWGN channels.

The rest of the chapter is organized as follows. We start with some preliminaries in Section 4.2. Section 4.3 discusses the characterization of GASTs through their WCMs, in addition to the optimality proof and the WCMs enumeration. In Section 4.4 we detail our analysis for the process of the GAST removal through WCMs. Afterwards, Section 4.5 discusses OSTs and how to customize the WCM framework to remove them. The simulation results are presented in Section 4.6. The chapter ends with concluding remarks in Section 4.7.

4.2 Preliminaries

The definitions of different objects of interest are provided in Chapter 2 and Chapter 3. In particular, see Definition 5 for GASs, Definition 6 for unlabeled GASs, Definition 7 for GASTs, Definition 3 for BASs, and Definition 2 for EASs. Moreover, the WCM framework is described in detail in Chapter 3.

Let γ be the column weight (VN degree) of the NB-LDPC code. Recall that $g = \left\lfloor \frac{\gamma-1}{2} \right\rfloor$.

GF refers to Galois field, and q is the GF size (order). We focus here on the case of $q = 2^{\lambda}$, where λ is a positive integer $\geqslant 2$. Furthermore, when we say in this chapter that nodes are "connected", we mean they are "directly connected" or they are "neighbors", unless otherwise stated. The same applies conceptually when we say an edge is "connected" to a node or vice versa.

In this chapter, all vectors are column vectors, except the cutting vectors of SC codes and the equalization target of the PR channel (see Subsection 4.6.4). Furthermore, in all GAST, unlabeled GAST, and OST figures, circles represent VNs. In all GAST and OST figures, grey (resp., white) squares represent unsatisfied (resp., satisfied) CNs. In all unlabeled GAST figures, grey (resp., white) squares represent degree-1 (resp., > 1) CNs.

The three theorems essential for understanding the WCM framework are Theorem 1, Theorem 2, and Theorem 3 in Chapter 3. Recall that given an (a, d_1, d_2, d_3) unlabeled GAST, the maximum number of unsatisfied CNs in the resulting GAST after edge labeling, b_{max}, is upper bounded by $d_1 + b_{\text{ut}}$. Here, b_{ut} is the upper bound on the maximum number of degree-2 unsatisfied CNs the resulting GAST can have. Because of the structure of the underlying unlabeled configuration, sometimes the exact maximum (obtained by Algorithm 2) is a quantity smaller than b_{ut}. We refer to this exact maximum as b_{et}. Thus,

$$b_{\text{max}} = d_1 + b_{\text{et}}. \tag{4.1}$$

We also recall the following. For a given GAST, a matrix $\mathbf{W}^{\text{z}}$ is defined as the matrix obtained by removing b', $d_1 \leqslant b' \leqslant b_{\text{max}}$, rows corresponding to CNs having either degree 1 or degree 2 from the matrix $\mathbf{A}$, the GAST adjacency matrix. These b' CNs can simultaneously be unsatisfied under some edge labeling that produces a GAST which has the same unlabeled GAST as the given GAST. Let $\mathcal{U}$ be the set of all such matrices $\mathbf{W}^{\text{z}}$.

The definition of the GAST removal is Definition 8. WCMs are then described in Definition 9. Recall also that each matrix $\mathbf{W}^{\text{z}}$ contains at least one WCM as its submatrix.

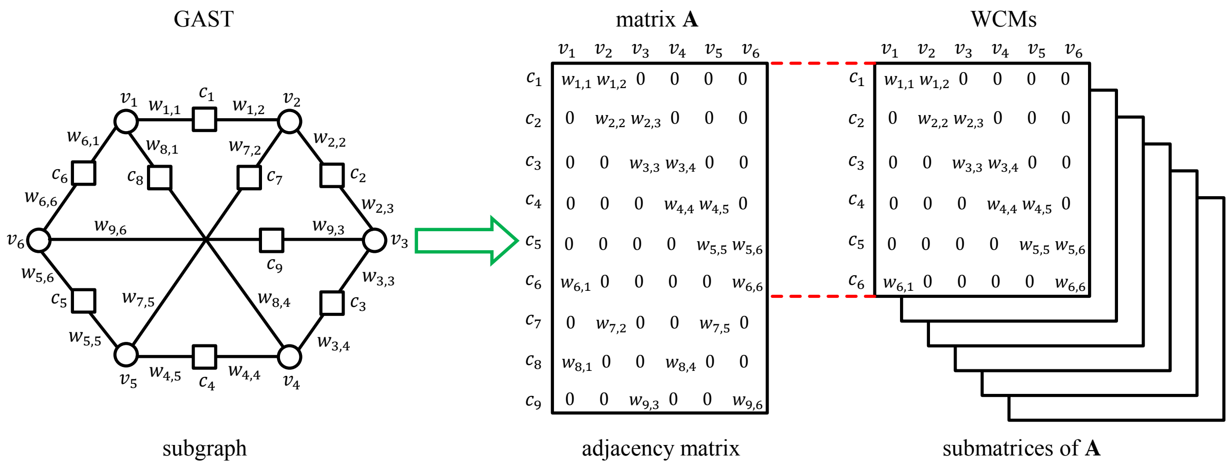


Figure 4.1: An illustrative figure showing the process of extracting the WCMs of a $(6, 0, 0, 9, 0)$ GAST. Appropriate edge weights (w's) $\in \text{GF}(q)\backslash\{0\}$ are assumed.

Theorem 3 then demonstrates how WCMs are used to remove a GAST from the Tanner graph of an NB-LDPC code. WCMs are denoted by $\mathbf{W}_h^{\text{cm}}$, $1 \leqslant h \leqslant t$, with size $(\ell - b_h^{\text{cm}}) \times a$, and the set of all WCMs is the set $\mathcal{W}$.

Definition 13. *Parameter b_{et} represents the **exact maximum number** of rows corresponding to degree-2 CNs that can be removed together from $\mathbf{A}$ to extract a WCM. Similarly, we define b_{st} to be the **exact minimum number** of rows corresponding to degree-2 CNs that can be removed together from $\mathbf{A}$ to extract a WCM. Recall that the rows corresponding to degree-1 CNs are always removed while extracting a WCM. Thus, $d_1 \leqslant d_1 + b_{\text{st}} \leqslant b_h^{\text{cm}} \leqslant d_1 + b_{\text{et}} = b_{\text{max}}$. Both b_{st} and b_{et} depend on the unlabeled GAST.*

Fig. 4.1 depicts the relation between a GAST and its associated WCMs, and roughly describes how the WCMs of this GAST are extracted.

The WCM framework is easily adjusted to efficiently remove special subclasses of GASTs, namely EASs and BASTs, by customizing the WCM definition. The details are in Chapter 3; particularly, see Lemma 4 in addition to Definitions 10, 11, and 12.

The two algorithms that constitute the WCM framework are Algorithm 2, which is the WCM extraction algorithm, and Algorithm 3, which is the code optimization algorithm. The two algorithms are also in Chapter 3.

A WCM that has (3.8) satisfied is said to be a WCM with **broken weight conditions**. A GAST is removed if and only if all its WCMs have broken weight conditions. Note that the complexity of the process of removing a specific GAST using the WCM framework is mainly controlled by the number of WCMs, which is t, of that GAST (see the **for** loop in Step 12 of Algorithm 3). Thus, the complexity of the WCM framework depends on the size of the set $\mathcal{G}$ (see Algorithm 3) and the numbers of WCMs of the GASTs in $\mathcal{G}$.

4.3 Characterizing GASTs Through Their WCMs

In order to characterize a GAST through its WCMs, we introduce the definition of the GAST tree, which will also be used to derive all the results in this section. Since this tree does not depend on the edge weights of the configuration, we call it the unlabeled GAST tree.

Recall that $\mathbf{A}$ is the adjacency matrix of the GAST. Both $\mathbf{W}^{\mathrm{z}}$ and $\mathcal{U}$ are defined in Section 4.2. Recall also that b_{et} is the maximum number of degree-2 CNs that can be unsatisfied simultaneously while the object remains a GAST. As in Chapter 3, define u^0 as the number of degree-2 CNs that can be unsatisfied individually while the object remains a GAST, and $\mathbf{y}^0$ as the vector in which the indices of such u^0 CNs are saved. Note that we always have $b_{\mathrm{et}} \leqslant u^0$.

Definition 14. *For a given (a, d_1, d_2, d_3) unlabeled GAST with $b_{\mathrm{et}} > 0$, we construct the **unlabeled GAST tree** of b_{et} levels (level 0 is not counted) as follows:*

- *Except the root node at level 0, each tree node represents a degree-2 CN in the unlabeled GAST. For any two CNs in the tree, being neighbors means that they can be unsatisfied simultaneously after labeling and the resulting object remains a GAST.*

- *Let $i_1, i_2, \dots, i_{b_{\mathrm{et}}}$ be the running indices used to access nodes at different levels in the tree as follows. The index of a node at level j, $1 \leqslant j \leqslant b_{\mathrm{et}}$, is saved in $\mathbf{y}^{j-1}_{i_1,i_2,\dots,i_{j-1}}$ and given by $y^{j-1}_{i_1,i_2,\dots,i_{j-1}}(i_j)$. CN $c_{y^{j-1}_{i_1,i_2,\dots,i_{j-1}}(i_j)}$ at level j is accessed via the path of nodes "root node – $c_{y^0(i_1)}$ – $c_{y^1_{i_1}(i_2)}$ – $c_{y^2_{i_1,i_2}(i_3)}$ – $\dots$ – $c_{y^{j-1}_{i_1,i_2,\dots,i_{j-1}}(i_j)}$".*

- *At level* 0*, a virtual root node is assumed to be connected to the* u^0 *nodes with indices in* $\mathbf{y}^0$ *at level* 1*. Level* j *of the tree consists of all the nodes with indices in* $\mathbf{y}^{j-1}_{i_1,i_2,\dots,i_{j-1}}$, $\forall i_1, i_2, \dots, i_{j-1}$*. Level* $j+1$ *of the tree is created as follows. Each CN* $c_{y^{j-1}_{i_1,i_2,\dots,i_{j-1}}(i_j)}$ *at level* j *is connected to all the CNs with indices in* $\mathbf{y}^{j}_{i_1,i_2,\dots,i_j}$ *at level* $j+1$*. These CNs can each be – simultaneously with the nodes on the path from the root node until* $c_{y^{j-1}_{i_1,i_2,\dots,i_{j-1}}(i_j)}$ *– unsatisfied after labeling and the resulting object remains a GAST.*

- *The number of nodes at level* $j+1$ *that are connected to* $c_{y^{j-1}_{i_1,i_2,\dots,i_{j-1}}(i_j)}$ *at level* j *is* $u^{j}_{i_1,i_2,\dots,i_j}$ *(which is the size of the vector* $\mathbf{y}^{j}_{i_1,i_2,\dots,i_j}$*), with* $u^{j}_{i_1,i_2,\dots,i_j} < u^{j-1}_{i_1,i_2,\dots,i_{j-1}}, \forall i_1, i_2, \dots, i_j$*.*

- *The leaves of this tree are linked to the matrices extracted by Algorithm 2 before removing the repeated matrices (see Algorithm 2).*

Note that for the parameters u and $\mathbf{y}$, the superscript refers to the level prior to the level in which the nodes exist, and the subscript refers to the running indices used to access the nodes. Note also that Algorithm 2 is designed to generate the unlabeled GAST tree.

Fig. 4.2 shows an unlabeled GAST tree for a configuration that has $b_{\text{et}} = 3$. The configuration has three levels after the root node. We say that each tree node at level j, $j > 0$, in the unlabeled GAST tree is ***linked to*** a matrix $\mathbf{W}^{\text{z}} \in \mathcal{U}$ extracted by removing $(d_1 + j)$ rows from the matrix $\mathbf{A}$. These rows correspond to all the d_1 degree-1 CNs, and the j degree-2 CNs on the path from the virtual root node to this tree node in the configuration. We also say that every valid matrix $\mathbf{W}^{\text{z}} \in \mathcal{U}$ is ***linked to*** one or more tree nodes.

It can be shown that b_{et} is the number of levels (nested loops in Algorithm 2), after which $u^{b_{\text{et}}}_{i_1,i_2,\dots,i_{b_{\text{et}}}} = 0$, $\forall i_1, i_2, \dots, i_{b_{\text{et}}}$. Moreover, because the WCMs do not necessarily have the same row dimension, Algorithm 2 may stop at b_{k} levels, $b_{\text{k}} \leqslant b_{\text{et}}$, starting from some $c_{y^0(i_1)}$, which results in an $(\ell - b^{\text{cm}}_h) \times a$ WCM with $b^{\text{cm}}_h = d_1 + b_{\text{k}} \leqslant b_{\text{max}} = d_1 + b_{\text{et}}$. The smallest value of b_{k} is b_{st}, i.e., $b_{\text{st}} \leqslant b_{\text{k}} \leqslant b_{\text{et}}$.

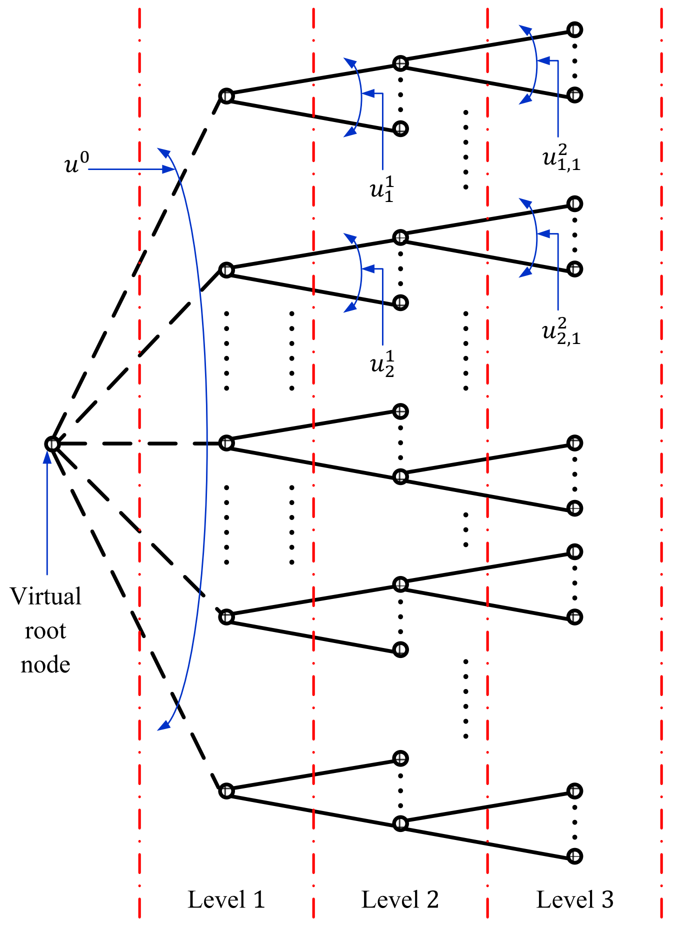


Figure 4.2: An unlabeled GAST tree with $b_{\text{et}} = 3$.

Remark 15. *Note that the unlabeled GAST tree is unique for a given unlabeled configuration. In other words, two non-isomorphic* (a, d_1, d_2, d_3) *configurations have two different unlabeled GAST trees even though they have the same* a*,* d_1*,* d_2*, and* d_3 *parameters.*

Repetitions in tree nodes linked to matrices $\mathbf{W}^{\text{z}}$ come from the fact that we are addressing the permutations and not the combinations in the tree. In other words, if we have a path from the root node at level 0 to a tree node at level 2 that has c_1 at level 1 then c_4 at level 2 on it, there must exist another path from the root node at level 0 to another tree node at level 2 that has c_4 at level 1 then c_1 at level 2 on it. Obviously, removing the row of c_1 then the row of c_4, or first c_4 then c_1 (in addition to the rows of degree-1 CNs) from $\mathbf{A}$ to extract a matrix produces the same end result.

4.3.1 Proving the Optimality of the WCM Framework

The numbers of matrices needed to operate on for different GASTs control the complexity of the code optimization process. In this subsection, we prove that the WCM framework is optimal in the sense that it works on the minimum possible number of matrices to remove a GAST. Our optimization problem is formulated as follows:

The optimization problem: We seek to find the set $\mathcal{W}$ of matrices that has the minimum cardinality, with the matrices in $\mathcal{W}$ representing submatrices of $\mathbf{A}$ that can be used to remove the problematic GAST, without the need to work on other submatrices.

The optimization constraint: Each matrix in $\mathcal{W}$ has to be a valid $\mathbf{W}^{\mathrm{z}}$ matrix in $\mathcal{U}$.

The optimization constraint is set to ensure that we are performing not only sufficient, but also necessary processing to remove the object. Note that, by definition, the set of WCMs is the solution of this optimization problem. Thus, the problem of proving the optimality of the WCM framework reduces to proving that the matrices we extract by Algorithm 2, and operate on in Algorithm 3 to remove the GAST, are indeed the WCMs.

Now, we are ready to present the optimality theorem and its proof.

Theorem 4. *Consider an (a, b, d_1, d_2, d_3) GAST with $b_{\mathrm{et}} > 0$. After eliminating repetitions, the set of matrices which are linked to the leaves of the unlabeled GAST tree characterized by Definition 14 is the set of WCMs, i.e., the set $\mathcal{W}$ of minimum cardinality.*

Proof. According to Definition 9, Theorem 3, and its proof, each matrix $\mathbf{W}^{\mathrm{z}} \in \mathcal{U}$ must have at least one matrix $\in \mathcal{W}$ as its submatrix (as $\mathcal{W}$ is the set of WCMs). The relation between a matrix $\mathbf{W}^{\mathrm{z}}_1$ linked to tree node 1 at level j and a matrix $\mathbf{W}^{\mathrm{z}}_2$ linked to tree node 2 at level $j + 1$, provided that tree node 2 is a child of tree node 1, is as follows. Matrix $\mathbf{W}^{\mathrm{z}}_2$ is a submatrix of matrix $\mathbf{W}^{\mathrm{z}}_1$, extracted by removing one more row, that corresponds to a degree-2 CN (tree node 2), from $\mathbf{W}^{\mathrm{z}}_1$. Following the same logic, the multiset of matrices, say $\mathcal{W}_{\mathrm{rep}}$, linked to tree nodes with no children (the leaves of the tree) contains submatrices of every possible matrix $\mathbf{W}^{\mathrm{z}}$. We let the set $\mathcal{W}_{\mathrm{nrep}}$ be $\mathcal{W}_{\mathrm{rep}}$ after eliminating the repetitions.

Now, we prove the sufficiency and minimality, which imply the optimality of $\mathcal{W}_{\text{nrep}}$. The sufficiency is proved as follows. Any matrix $\mathbf{W}^{\text{z}}$ that is linked to a tree node with a child will be redundant if added to the set $\mathcal{W}_{\text{nrep}}$ because in $\mathcal{W}_{\text{nrep}}$ there already exists a submatrix of this $\mathbf{W}^{\text{z}}$ (from the analysis above). The minimality is proved as follows. If we eliminate any matrix from $\mathcal{W}_{\text{nrep}}$, there will be at least one matrix $\mathbf{W}^{\text{z}}$ that has no submatrices in $\mathcal{W}_{\text{nrep}}$ (which is the eliminated matrix itself since it is linked to a node (nodes) with no children). Thus, we cannot further reduce the cardinality of $\mathcal{W}_{\text{nrep}}$. Hence, the set $\mathcal{W}_{\text{nrep}}$ is indeed the set $\mathcal{W}$ of WCMs, which proves the optimality of the WCM framework. □

4.3.2 Enumeration of WCMs Associated with a GAST

In this subsection, we provide the exact number of distinct WCMs associated with a GAST. Moreover, we present particular examples where this number reduces to a combinatorial function of the column weight of the code. Since the number of WCMs (and also their sizes) associated with a GAST only depends on the unlabeled configuration and not on the edge weights, we relate the number of distinct WCMs, t, to the unlabeled GAST throughout this chapter.

We first identify the following two types of unlabeled GAST configurations according to the properties of their unlabeled GAST trees.

Definition 15. *An (a, d_1, d_2, d_3) **same-size-WCMs** unlabeled GAST satisfies one of the following two conditions:*

1. *It has $b_{\text{et}} = 0$, i.e., $u^0 = 0$, which results in $|\mathcal{W}| = 1$.*
2. *It has $b_{\text{et}} > 0$; thus, $u^0 > 0$, and its tree has the property that $u^j_{i_1,i_2,\dots,i_j} = 0$ only if $j = b_{\text{et}}$, $\forall i_1, i_2, \dots, i_{b_{\text{et}}}$, which results in all the WCMs having the same $(\ell - b_{\text{max}}) \times a$ size, $b_{\text{max}} = d_1 + b_{\text{et}}$.*

Definition 16. *An (a, d_1, d_2, d_3) **u-symmetric** unlabeled GAST is a same-size-WCMs unlabeled GAST which satisfies the following condition. If $u^0 > 0$, its tree has the property that* *at any level j, $u^{j-1}_{i_1,i_2,\dots,i_{j-1}}$ is the same, $\forall i_1, i_2, \dots, i_{j-1}$.*

An example of a same-size-WCMs unlabeled GAST that is not u-symmetric is the $(7, 9, 13, 0)$ configuration shown in Fig. 4.4(a). The $(6, 0, 9, 0)$ and the $(8, 0, 16, 0)$ configurations shown in Fig. 4.6(a) are examples of u-symmetric unlabeled GASTs.

We start off with the count for the general case.

Theorem 5. *Given the unlabeled GAST tree, an (a, d_1, d_2, d_3) unlabeled GAST, with the parameters $b_{\text{st}} > 0$ and $b_{\text{et}} > 0$, results in the following number, t, of distinct WCMs (t is the size of the set $\mathcal{W}$) for the labeled configuration:*

$$t = \sum_{b_\text{k}=b_\text{st}}^{b_\text{et}} \frac{1}{b_\text{k}!} \sum_{i_1=1}^{u^0} \sum_{i_2=1}^{u^1_{i_1}} \sum_{i_3=1}^{u^2_{i_1,i_2}} \cdots \sum_{i_{b_\text{k}}=1}^{u^{b_\text{k}-1}_{i_1,i_2,\dots,i_{b_\text{k}-1}}} T\left(u^{b_\text{k}}_{i_1,i_2,\dots,i_{b_\text{k}}}\right), \tag{4.2}$$

where $b_\text{st} \leqslant b_\text{k} \leqslant b_\text{et}$. Here, $T\left(u^{b_\text{k}}_{i_1,i_2,\dots,i_{b_\text{k}}}\right) = 1$ if $u^{b_\text{k}}_{i_1,i_2,\dots,i_{b_\text{k}}} = 0$, and $T\left(u^{b_\text{k}}_{i_1,i_2,\dots,i_{b_\text{k}}}\right) = 0$ otherwise.

Proof. To prove Theorem 5, we recall the unlabeled GAST tree. The number of nodes in this tree at any level $b_\text{k} > 0$ is given by:

$$\mu_{b_\text{k}} = \sum_{i_1=1}^{u^0} \sum_{i_2=1}^{u^1_{i_1}} \sum_{i_3=1}^{u^2_{i_1,i_2}} \cdots \sum_{i_{b_\text{k}}=1}^{u^{b_\text{k}-1}_{i_1,i_2,\dots,i_{b_\text{k}-1}}} (1). \tag{4.3}$$

From the previous subsection, the number, $t_{\text{rep},b_\text{k}}$, of WCMs (not necessarily distinct) extracted by removing $b^{\text{cm}}_h = d_1 + b_\text{k}$ rows from $\mathbf{A}$ equals the number of leaves at level b_k. Note that the leaves at level b_k do not have connections to level $b_\text{k} + 1$ (no children) in the tree. As a result, $t_{\text{rep},b_\text{k}}$ is given by:

$$t_{\text{rep},b_\text{k}} = \sum_{i_1=1}^{u^0} \sum_{i_2=1}^{u^1_{i_1}} \sum_{i_3=1}^{u^2_{i_1,i_2}} \cdots \sum_{i_{b_\text{k}}=1}^{u^{b_\text{k}-1}_{i_1,i_2,\dots,i_{b_\text{k}-1}}} T\left(u^{b_\text{k}}_{i_1,i_2,\dots,i_{b_\text{k}}}\right). \tag{4.4}$$

To compute the number of distinct WCMs, we need to eliminate repeated WCMs. Since

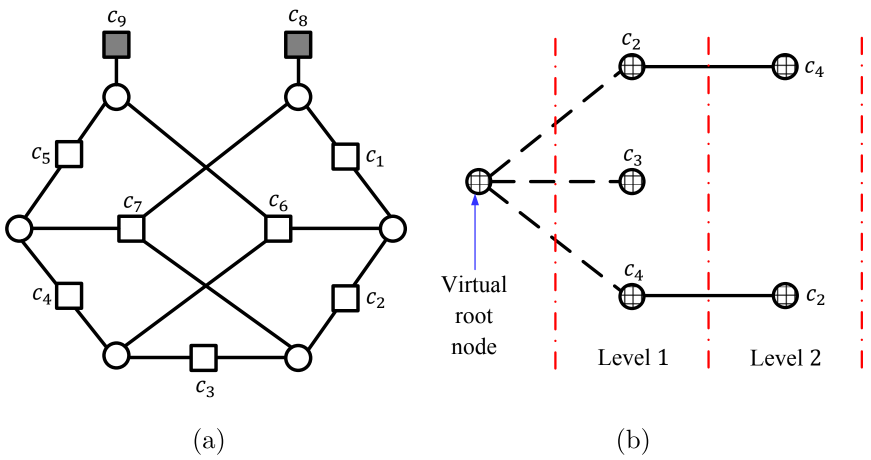


Figure 4.3: (a) A $(6, 2, 5, 2)$ unlabeled GAST for $\gamma = 3$. (b) The associated unlabeled GAST tree with $b_{\text{et}} = 2$.

a WCM extracted by removing $(d_1 + b_{\text{k}})$ rows from $\mathbf{A}$ appears $b_{\text{k}}!$ times, we compute the number of distinct WCMs that are extracted by removing $(d_1 + b_{\text{k}})$ rows from $\mathbf{A}$ using (4.4) as follows:

$$t_{b_{\text{k}}} = \frac{1}{b_{\text{k}}!} \sum_{i_1=1}^{u^0} \sum_{i_2=1}^{u^1_{i_1}} \sum_{i_3=1}^{u^2_{i_1,i_2}} \cdots \sum_{i_{b_{\text{k}}}=1}^{u^{b_{\text{k}}-1}_{i_1,i_2,\dots,i_{b_{\text{k}}-1}}} T\left(u^{b_{\text{k}}}_{i_1,i_2,\dots,i_{b_{\text{k}}}}\right). \tag{4.5}$$

The total number of distinct WCMs is then obtained by summing $t_{b_{\text{k}}}$ in (4.5) over all values of b_{k}, $b_{\text{st}} \leqslant b_{\text{k}} \leqslant b_{\text{et}}$, to reach t in (4.2). □

Recall that γ is the column weight (VN degree) of the code.

Example 12. *Fig. 4.3(a) shows a $(6, 2, 5, 2)$ unlabeled GAST for $\gamma = 3$. As demonstrated by the unlabeled GAST tree in Fig. 4.3(b), the configuration has WCMs that are not of the same size. Since $b_{\text{st}} = 1$, $b_{\text{et}} = b_{\text{ut}} = 2$, and $u^0 = 3$ (that are c_2, c_3, and c_4), (4.2) reduces to:*

$$t = \sum_{b_{\text{k}}=1}^{2} \frac{1}{b_{\text{k}}!} \sum_{i_1=1}^{3} \sum_{i_2=1}^{u^1_{i_1}} T\left(u^{b_{\text{k}}}_{i_1,\dots,i_{b_{\text{k}}}}\right)$$
$$= \frac{1}{1!}(0 + 1 + 0) + \frac{1}{2!}(1 + 0 + 1) = 2.$$

Thus, the configuration has only 2 WCMs, extracted by removing the rows of the following groups of CNs from $\mathbf{A}$: $\{(c_3, \mathcal{O}_{\text{sg}}), (c_2, c_4, \mathcal{O}_{\text{sg}})\}$, where $\mathcal{O}_{\text{sg}}$ is (c_8, c_9). We explicitly list the *subgroup $\mathcal{O}_{\text{sg}}$ of degree-1 CNs to highlight the fact that the rows of these CNs are always removed, irrespective of the action on the remaining rows in* $\mathbf{A}$.

Now, we analyze the important special case of same-size-WCMs configurations.

Lemma 6. *Given the unlabeled GAST tree, a same-size-WCMs (a, d_1, d_2, d_3) unlabeled GAST, with the parameter $b_{\text{et}} > 0$, results in the following number, t, of distinct WCMs (t is the size of the set $\mathcal{W}$) for the labeled configuration:*

$$t = \frac{1}{b_{\text{et}}!} \sum_{i_1=1}^{u^0} \sum_{i_2=1}^{u^1_{i_1}} \sum_{i_3=1}^{u^2_{i_1,i_2}} \cdots \sum_{i_{b_{\text{et}}}=1}^{u^{b_{\text{et}}-1}_{i_1,i_2,\dots,i_{b_{\text{et}}-1}}} (1)\,. \tag{4.6}$$

Proof. We prove Lemma 6 by substituting $b_{\text{k}} = b_{\text{st}} = b_{\text{et}}$ in (4.2):

$$\begin{aligned} t &= \frac{1}{b_{\text{et}}!} \sum_{i_1=1}^{u^0} \sum_{i_2=1}^{u^1_{i_1}} \sum_{i_3=1}^{u^2_{i_1,i_2}} \cdots \sum_{i_{b_{\text{et}}}=1}^{u^{b_{\text{et}}-1}_{i_1,i_2,\dots,i_{b_{\text{et}}-1}}} T\left(u^{b_{\text{et}}}_{i_1,i_2,\dots,i_{b_{\text{et}}}}\right) \\ &= \frac{1}{b_{\text{et}}!} \sum_{i_1=1}^{u^0} \sum_{i_2=1}^{u^1_{i_1}} \sum_{i_3=1}^{u^2_{i_1,i_2}} \cdots \sum_{i_{b_{\text{et}}}=1}^{u^{b_{\text{et}}-1}_{i_1,i_2,\dots,i_{b_{\text{et}}-1}}} (1)\,. \end{aligned} \tag{4.7}$$

The second equality in (4.7) follows from the fact that $T\left(u^{b_{\text{et}}}_{i_1,i_2,\dots,i_{b_{\text{et}}}}\right) = 1$ since $u^{b_{\text{et}}}_{i_1,i_2,\dots,i_{b_{\text{et}}}} = 0$, $\forall i_1, i_2, \dots, i_{b_{\text{et}}}$, from the definition of b_{et}. □

Example 13. *Fig. 4.4(a) shows a $(7, 9, 13, 0)$ unlabeled GAST for $\gamma = 5$. As demonstrated by the unlabeled GAST tree in Fig. 4.4(b), this is a same-size-WCMs configuration. Since $b_{\text{et}} = b_{\text{ut}} = 2$ and $u^0 = 5$ (that are c_3, c_4, c_9, c_{11}, and c_{12}), (4.6) reduces to:*

$$t = \frac{1}{2!} \sum_{i_1=1}^{5} \sum_{i_2=1}^{u^1_{i_1}} (1) = \frac{1}{2}(1 + 1 + 3 + 2 + 3) = 5.$$

Thus, the configuration has 5 WCMs, all of the same size (11×7), extracted by removing the rows of the following groups of CNs from the matrix $\mathbf{A}$*:* $\{(c_3, c_{12}, \mathcal{O}_{\text{sg}}), (c_4, c_9, \mathcal{O}_{\text{sg}}), (c_9, c_{11}, \mathcal{O}_{\text{sg}}), (c_9, c_{12}, \mathcal{O}_{\text{sg}}), (c_{11}, c_{12}, \mathcal{O}_{\text{sg}})\}$*, where $\mathcal{O}_{\text{sg}}$ is $(c_{14}, c_{15}, c_{16}, c_{17}, c_{18}, c_{19}, c_{20}, c_{21}, c_{22})$.*

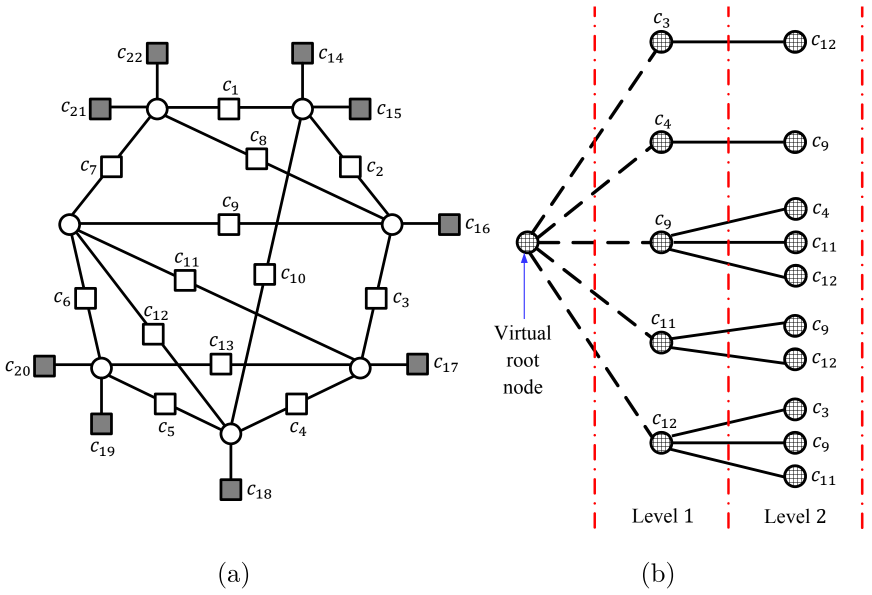


Figure 4.4: (a) A $(7, 9, 13, 0)$ unlabeled GAST for $\gamma = 5$. (b) The associated unlabeled GAST tree with $b_{\text{et}} = 2$.

Another important special case to study is the case of u-symmetric configurations.

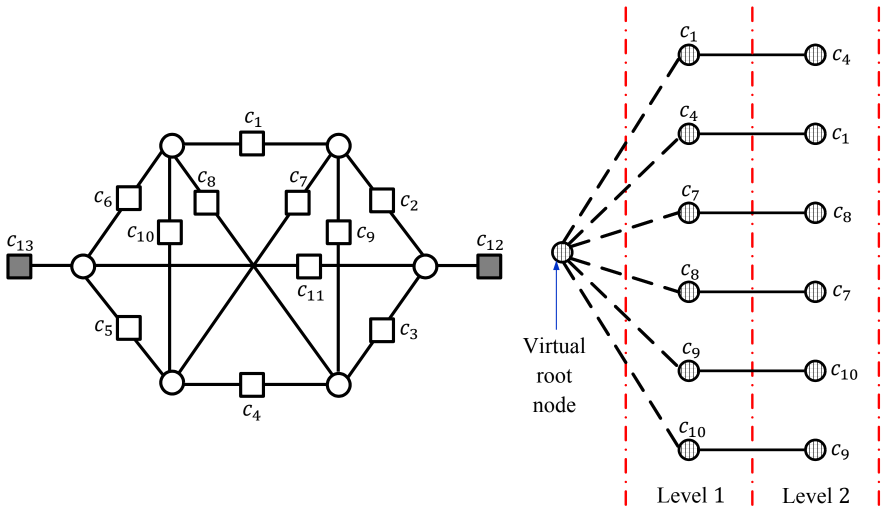
Corollary 1. *Given the unlabeled GAST tree, a u-symmetric* (a, d_1, d_2, d_3) *unlabeled GAST, with the parameter* $b_{\text{et}} > 0$*, results in the following number,* t*, of distinct WCMs (*t *is the size of the set* $\mathcal{W}$*) for the labeled configuration:*

$$t = \frac{1}{b_{\text{et}}!} \prod_{j=1}^{b_{\text{et}}} u^{j-1}. \tag{4.8}$$

Proof. Since the u-symmetric case is a special case of the same-size-WCMs case, we use (4.6) to conclude:

$$t = \frac{1}{b_{\text{et}}!} \sum_{i_1=1}^{u^0} \sum_{i_2=1}^{u^1} \sum_{i_3=1}^{u^2} \cdots \sum_{i_{b_{\text{et}}}=1}^{u^{b_{\text{et}}-1}} (1) = \frac{1}{b_{\text{et}}!} \prod_{j=1}^{b_{\text{et}}} u^{j-1}. \tag{4.9}$$

Equation (4.9) follows from the fact that for a u-symmetric configuration, at any level j, $u^{j-1}_{i_1,i_2,\dots,i_{j-1}}$ is the same, $\forall i_1, i_2, \dots, i_{j-1}$. Thus, we can express $u^{j-1}_{i_1,i_2,\dots,i_{j-1}}$ in (4.6) as u^{j-1}, which is independent of $i_1, i_2, \dots, i_{b_{\text{et}}-1}$, $\forall j \in \{1, 2, \dots, b_{\text{et}}\}$. $\square$



(a) (b)

Figure 4.5: (a) A $(6, 2, 11, 0)$ unlabeled GAST for $\gamma = 4$. (b) The associated unlabeled GAST tree with $b_{\text{et}} = 2$.

Example 14. *Fig. 4.5(a) shows a* $(6, 2, 11, 0)$ *unlabeled GAST for* $\gamma = 4$*. As demonstrated by the unlabeled GAST tree in Fig. 4.5(b), the configuration is u-symmetric. Since* $b_{\text{et}} = b_{\text{ut}} = 2$*,* $u^0 = 6$*, and* $u^1 = 1$*, (4.8) reduces to:*

$$t = \frac{1}{2!} \prod_{j=1}^{2} u^{j-1} = \frac{1}{2}(6)(1) = 3.$$

Thus, the configuration has 3 WCMs, all of the same size (9×6)*, extracted by removing the rows of the following groups of CNs from the matrix* $\mathbf{A}$*:* $\{(c_1, c_4, \mathcal{O}_{\text{sg}}), (c_7, c_8, \mathcal{O}_{\text{sg}}), (c_9, c_{10}, \mathcal{O}_{\text{sg}})\}$*, where* $\mathcal{O}_{\text{sg}}$ *is* (c_{12}, c_{13})*.*

After providing the exact number of WCMs for different cases, we now study examples where the number of distinct WCMs associated with a configuration is proved to be a function only of the column weight γ (the VN degree). We study the u-symmetric version of the $(2\gamma, 0, \gamma^2, 0)$ unlabeled GASTs with $g = \left\lfloor \frac{\gamma-1}{2} \right\rfloor = 1$ (i.e., for $\gamma = 3$ or $\gamma = 4$). Studying these configurations is important because they are unlabeled low weight codewords, and their multiplicity in the Tanner graph of a code typically strongly affects the error floor (and also the waterfall) performance of this code.

Lemma 7. *A u-symmetric $(2\gamma, 0, \gamma^2, 0)$ unlabeled GAST, with $\gamma \in \{3, 4\}$ (see Fig. 4.6(a)), results in $t = \gamma!$ distinct WCMs for the labeled configuration.*

Proof. From (3.5), for a u-symmetric $(2\gamma, 0, \gamma^2, 0)$ unlabeled GAST[2], we have:

$$b_{\text{et}} = b_{\text{ut}} = \left\lfloor \frac{1}{2} \left(2\gamma \left\lfloor \frac{\gamma - 1}{2} \right\rfloor - 0 \right) \right\rfloor = \gamma. \tag{4.10}$$

Notice that $\left\lfloor \frac{\gamma-1}{2} \right\rfloor = 1$ for $\gamma \in \{3, 4\}$. Substituting (4.10) in (4.8) gives:

$$t = \frac{1}{\gamma!} \prod_{j=1}^{\gamma} u^{j-1} = \frac{\gamma^2}{\gamma!} \prod_{j=2}^{\gamma} u^{j-1}, \tag{4.11}$$

where the second equality in (4.11) follows from the property that for a $(2\gamma, 0, \gamma^2, 0)$ unlabeled GAST, $u^0 = \gamma^2$.

Next, we compute u^{j-1}, $2 \leqslant j \leqslant b_{\text{et}} = \gamma$. At level 1, a degree-2 CN that has its index in $\mathbf{y}^0$ will be marked as unsatisfied resulting in:

$$u^1 = u^0 - 1 - 2(\gamma - 1) = \gamma^2 - 2\gamma + 1 = (\gamma - 1)^2. \tag{4.12}$$

Equation (4.12) follows from the fact that after such a degree-2 CN is selected to be marked as unsatisfied at level 1, all the remaining $(\gamma - 1)$ CNs connected to each of the two VNs sharing this CN cannot be selected at level 2 (because $g = 1$ for $\gamma \in \{3, 4\}$). Thus, $u^1 = u^0 - (1+2(\gamma - 1))$, where the additional 1 represents the already selected CN itself. Furthermore:

$$u^2 = u^1 - 1 - 2(\gamma - 2) = (\gamma - 1)^2 - 2\gamma + 3 = (\gamma - 2)^2. \tag{4.13}$$

Note that the $2(\gamma - 1)$ CNs that cannot be selected at level 2 are connected to all the remaining $(2\gamma - 2)$ VNs in the configuration (after excluding the two VNs sharing the CN selected at level 1). Thus, any CN to be selected at level 2 results in $2(\gamma - 2)$ extra CNs

[2]Unlike the superscript of u, the superscript of γ and all linear expressions of γ refers to the mathematical power (if exists).

that cannot be selected[3] at level 3. As a result, $u^2 = u^1 - (1 + 2(\gamma - 2))$, which is Equation (4.13). This analysis also applies for u^{j-1} with $j > 3$. By means of induction, we conclude the following for every u^{j-1} with $1 \leqslant j \leqslant b_{\text{et}} = \gamma$:

$$u^{j-1} = (\gamma - (j-1))^2 . \tag{4.14}$$

Substituting (4.14) into (4.11) gives:

$$t = \frac{1}{\gamma!}\gamma^2 (\gamma-1)^2 (\gamma-2)^2 \cdots 1^2 = \gamma!. \tag{4.15}$$

As a result, $t = \gamma!$, which completes the proof. $\square$

Example 15. *Fig. 4.6(a), upper panel, shows the u-symmetric* $(6, 0, 9, 0)$ *unlabeled GAST for* $\gamma = 3$*. Fig. 4.6(b) confirms that the configuration is u-symmetric with* $b_{\text{et}} = b_{\text{ut}} = 3$*,* $u^0 = 9$*,* $u^1 = 4$*, and* $u^2 = 1$*. Thus, (4.8) reduces to (4.15), implying:*

$$t = \frac{1}{3!}\prod_{j=1}^{3} u^{j-1} = 3! = 6.$$

The configuration has 6 *WCMs (size* 6×6*), extracted by removing the rows of the following groups of CNs from* $\mathbf{A}$*:* $\{(c_1, c_3, c_5), (c_1, c_4, c_9), (c_2, c_4, c_6), (c_2, c_5, c_8), (c_3, c_6, c_7), (c_7, c_8, c_9)\}$.

Fig. 4.6(a), lower panel, shows the u-symmetric $(8, 0, 16, 0)$ *unlabeled GAST for* $\gamma = 4$*. We omit the tree of this unlabeled GAST for brevity. Following the same logic we used for the u-symmetric* $(6, 0, 9, 0)$ *unlabeled GAST (*$\gamma = 3$*), we conclude that this configuration has* $b_{\text{et}} = 4$*,* $u^0 = 16$*,* $u^1 = 9$*,* $u^2 = 4$*, and* $u^3 = 1$*. Thus, from (4.15):*

$$t = \frac{1}{4!}\prod_{j=1}^{4} u^{j-1} = 4! = 24.$$

[3]The reason why it is $2(\gamma - 2)$ and not $2(\gamma - 1)$ is that two CNs from the group that cannot be selected at level 3 were already accounted for while computing u^1 as they could not be selected at level 2 (recall that the configuration is u-symmetric).

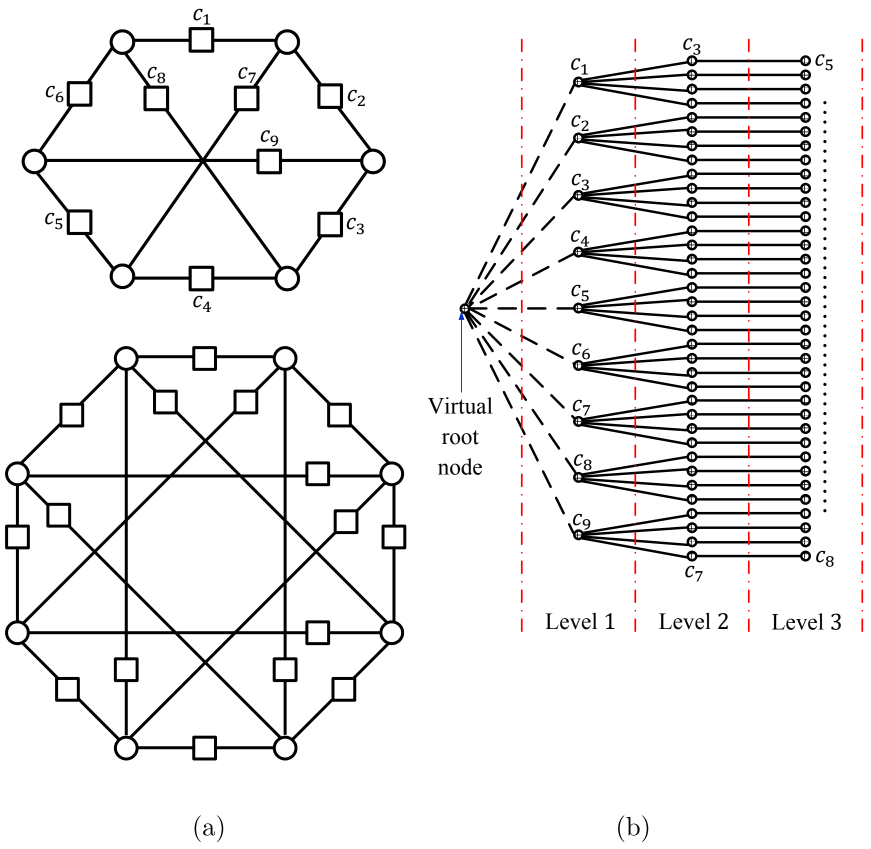


Figure 4.6: (a) Upper panel: the u-symmetric $(6, 0, 9, 0)$ unlabeled GAST for $\gamma = 3$. Lower panel: the u-symmetric $(8, 0, 16, 0)$ unlabeled GAST for $\gamma = 4$. (b) The associated unlabeled GAST tree for the $(6, 0, 9, 0)$ unlabeled GAST with $b_{\text{et}} = 3$.

Table 4.1: Number of distinct WCMs for different types of unlabeled GASTs.

Unlabeled GAST type	Number of distinct WCMs (t)
General	$t = \sum_{b_\text{k}=b_\text{st}}^{b_\text{et}} \frac{1}{b_\text{k}!} \sum_{i_1=1}^{u^0} \sum_{i_2=1}^{u^1_{i_1}} \sum_{i_3=1}^{u^2_{i_1,i_2}} \cdots \sum_{i_{b_\text{k}}=1}^{u^{b_\text{k}-1}_{i_1,i_2,\dots,i_{b_\text{k}-1}}} T\left(u^{b_\text{k}}_{i_1,i_2,\dots,i_{b_\text{k}}}\right)$
Same-size-WCMs	$t = \frac{1}{b_\text{et}!} \sum_{i_1=1}^{u^0} \sum_{i_2=1}^{u^1_{i_1}} \sum_{i_3=1}^{u^2_{i_1,i_2}} \cdots \sum_{i_{b_\text{et}}=1}^{u^{b_\text{et}-1}_{i_1,i_2,\dots,i_{b_\text{et}-1}}} (1)$
U-symmetric	$t = \frac{1}{b_\text{et}!} \prod_{j=1}^{b_\text{et}} u^{j-1}$
U-symmetric, $(2\gamma, 0, \gamma^2, 0)$, with $\gamma \in \{3, 4\}$	$t = \gamma!$

We conclude Subsection 4.3.2 with Table 4.1. Table 4.1 lists the number of distinct WCMs for different types of unlabeled GASTs.

4.3.3 Complexity Comparison with a Suboptimal Idea

We have already proved the optimality of the WCM framework in Subsection 4.3.1. In this subsection, we demonstrate the complexity reduction we gain by focusing only on the set of WCMs, $\mathcal{W}$, to remove a GAST. We compute the total number of distinct matrices to operate on in an alternative idea (a suboptimal idea), and compare it with the number of distinct WCMs we operate on, which is t derived in Subsection 4.3.2. The suboptimal idea we compare with is operating on the set of all distinct matrices $\mathbf{W}^{\mathrm{z}}$.

Computational savings of the WCM framework relative to the suboptimal idea mentioned above on a prototypical example of an NB-LDPC code are quite apparent; it takes roughly only four days to optimize this code using the WCM framework (via operating on the WCMs of each GAST to be removed), compared with roughly a month of computations using the suboptimal approach (via operating on all distinct matrices $\mathbf{W}^{\mathrm{z}}$ of each GAST to be removed). In this subsection, we justify this observation.

Here, we seek to compare the number of distinct WCMs, which is the size of the set $\mathcal{W}$, with the number of distinct $\mathbf{W}^{\mathrm{z}}$ matrices, which is the size of the set $\mathcal{U}$. For convenience, we assume for this comparison that $b_{\mathrm{et}} > 0$ and $u^0 > 0$.

Theorem 6. *Given the unlabeled GAST tree, the difference between the cardinalities of the sets $\mathcal{U}$ and $\mathcal{W}$ (the reduction in the number of matrices to operate on) for an (a, d_1, d_2, d_3) unlabeled GAST, with the parameters $b_{\mathrm{st}} > 0$ and $b_{\mathrm{et}} > 0$, is:*

$$t' - t = 1 + \sum_{j=1}^{b_{\mathrm{et}}-1} \frac{1}{j!} \sum_{i_1=1}^{u^0} \sum_{i_2=1}^{u^1_{i_1}} \sum_{i_3=1}^{u^2_{i_1,i_2}} \cdots \sum_{i_j=1}^{u^{j-1}_{i_1,i_2,\dots,i_{j-1}}} T_{\mathrm{c}}\left(u^j_{i_1,i_2,\dots,i_j}\right), \tag{4.16}$$

where $t' = |\mathcal{U}|$ and $t = |\mathcal{W}|$. Here, $T_{\mathrm{c}}\left(u^j_{i_1,i_2,\dots,i_j}\right) = 1$ if $u^j_{i_1,i_2,\dots,i_j} \neq 0$, and $T_{\mathrm{c}}\left(u^j_{i_1,i_2,\dots,i_j}\right) = 0$ otherwise.

Proof. Given that t (which is $|\mathcal{W}|$) is known from Subsection 4.3.2, we need to derive t' (which is $|\mathcal{U}|$). Since $\mathcal{U}$ is the set of all distinct matrices $\mathbf{W}^{\mathrm{z}}$, it follows that the cardinality of $\mathcal{U}$ is a function of the total number of nodes in the unlabeled GAST tree. Note that each node at level j in the unlabeled GAST tree is linked to a matrix $\mathbf{W}^{\mathrm{z}}$ (see the proof of Theorem 4). The total number of these tree nodes is:

$$\eta_{\mathrm{rep}} = \sum_{j=1}^{b_{\mathrm{et}}} \sum_{i_1=1}^{u^0} \sum_{i_2=1}^{u^1_{i_1}} \sum_{i_3=1}^{u^2_{i_1,i_2}} \cdots \sum_{i_j=1}^{u^{j-1}_{i_1,i_2,\dots,i_{j-1}}} (1) \,. \tag{4.17}$$

To remove the repeated $\mathbf{W}^{\mathrm{z}}$ matrices from that count, we need to divide the number of tree nodes at each level j by $j!$. Thus, we reach:

$$\eta = \sum_{j=1}^{b_{\mathrm{et}}} \frac{1}{j!} \sum_{i_1=1}^{u^0} \sum_{i_2=1}^{u^1_{i_1}} \sum_{i_3=1}^{u^2_{i_1,i_2}} \cdots \sum_{i_j=1}^{u^{j-1}_{i_1,i_2,\dots,i_{j-1}}} (1) \,. \tag{4.18}$$

The cardinality of the set $|\mathcal{U}|$, which is t', is then:

$$t' = 1 + \eta = 1 + \sum_{j=1}^{b_{\mathrm{et}}} \frac{1}{j!} \sum_{i_1=1}^{u^0} \sum_{i_2=1}^{u^1_{i_1}} \sum_{i_3=1}^{u^2_{i_1,i_2}} \cdots \sum_{i_j=1}^{u^{j-1}_{i_1,i_2,\dots,i_{j-1}}} (1) \,, \tag{4.19}$$

where the additional 1 is for the particular matrix $\mathbf{W}^{\mathrm{z}}$ extracted by removing d_1 rows from $\mathbf{A}$ corresponding to all degree-1 CNs in the configuration. Note that we can consider the virtual root node as the node linked to this particular $\mathbf{W}^{\mathrm{z}}$ matrix in the tree.

To compute $t' - t$, we subtract (4.2) from (4.19). Consequently,

$$\begin{aligned} t' - t = 1 &+ \sum_{j=1}^{b_{\mathrm{st}}-1} \frac{1}{j!} \sum_{i_1=1}^{u^0} \sum_{i_2=1}^{u^1_{i_1}} \sum_{i_3=1}^{u^2_{i_1,i_2}} \cdots \sum_{i_j=1}^{u^{j-1}_{i_1,i_2,\dots,i_{j-1}}} (1) \\ &+ \sum_{j=b_{\mathrm{st}}}^{b_{\mathrm{et}}} \frac{1}{j!} \sum_{i_1=1}^{u^0} \sum_{i_2=1}^{u^1_{i_1}} \sum_{i_3=1}^{u^2_{i_1,i_2}} \cdots \sum_{i_j=1}^{u^{j-1}_{i_1,i_2,\dots,i_{j-1}}} \left[1 - T\left(u^{j}_{i_1,i_2,\dots,i_j}\right)\right] . \end{aligned} \tag{4.20}$$

Thus, we complete the proof as follows:

$$t' - t = 1 + \sum_{j=1}^{b_{\mathrm{et}}} \frac{1}{j!} \sum_{i_1=1}^{u^0} \sum_{i_2=1}^{u^1_{i_1}} \sum_{i_3=1}^{u^2_{i_1,i_2}} \cdots \sum_{i_j=1}^{u^{j-1}_{i_1,i_2,\dots,i_{j-1}}} \left[1 - T\left(u^j_{i_1,i_2,\dots,i_j}\right)\right]$$

$$= 1 + \sum_{j=1}^{b_{\mathrm{et}}-1} \frac{1}{j!} \sum_{i_1=1}^{u^0} \sum_{i_2=1}^{u^1_{i_1}} \sum_{i_3=1}^{u^2_{i_1,i_2}} \cdots \sum_{i_j=1}^{u^{j-1}_{i_1,i_2,\dots,i_{j-1}}} T_{\mathrm{c}}\left(u^j_{i_1,i_2,\dots,i_j}\right). \qquad (4.21)$$

The first equality in (4.21) is derived by observing that $\left[1 - T\left(u^j_{i_1,i_2,\dots,i_j}\right)\right] = 1$ for $j \in \{1, 2, \dots, b_{\mathrm{st}}-1\}$. The second equality in (4.21) follows from that $T\left(u^j_{i_1,i_2,\dots,i_j}\right) = 1$ for $j = b_{\mathrm{et}}$, and $\left[1 - T\left(u^j_{i_1,i_2,\dots,i_j}\right)\right] = T_{\mathrm{c}}\left(u^j_{i_1,i_2,\dots,i_j}\right)$ (from the definitions of both T and T_{c}). We can simply consider $T_{\mathrm{c}}\left(u^j_{i_1,i_2,\dots,i_j}\right)$ as the complement function (binary inversion) of $T\left(u^j_{i_1,i_2,\dots,i_j}\right)$.

□

Example 16. *Consider the $(6, 2, 5, 2)$ unlabeled GAST ($\gamma = 3$) shown in Fig. 4.3(a). Since $b_{\mathrm{st}} = 1$, $b_{\mathrm{et}} = b_{\mathrm{ut}} = 2$, and $u^0 = 3$, and aided by the unlabeled GAST tree in Fig. 4.3(b), the complexity reduction (the reduction in the number of matrices to operate on) is (see (4.16)):*

$$t' - t = 1 + \frac{1}{1!} \sum_{i_1=1}^{3} T_{\mathrm{c}}\left(u^1_{i_1}\right) = 1 + (1 + 0 + 1) = 3.$$

In other words, the cardinality of the set $\mathcal{U}$ is $t' = 5$, while from Example 12, the cardinality of the set $\mathcal{W}$ (the number of distinct WCMs) is $t = 2$. Thus, the complexity reduction is 60% for this configuration.

Now, we study the case of same-size-WCMs unlabeled GASTs.

Lemma 8. *Given the unlabeled GAST tree, the difference between the cardinalities of the sets $\mathcal{U}$ and $\mathcal{W}$ (the reduction in the number of matrices to operate on) for a same-size-WCMs (a, d_1, d_2, d_3) unlabeled GAST, with the parameter $b_{\mathrm{et}} > 0$, is:*

$$t' - t = 1 + \sum_{j=1}^{b_{\mathrm{et}}-1} \frac{1}{j!} \sum_{i_1=1}^{u^0} \sum_{i_2=1}^{u^1_{i_1}} \sum_{i_3=1}^{u^2_{i_1,i_2}} \cdots \sum_{i_j=1}^{u^{j-1}_{i_1,i_2,\dots,i_{j-1}}} (1), \qquad (4.22)$$

where $t' = |\mathcal{U}|$ and $t = |\mathcal{W}|$.

Proof. Knowing that the configuration is a same-size-WCMs unlabeled GAST does not simplify the expression of t' in (4.19). Thus, to compute $(t' - t)$, we subtract (4.6) from (4.19). The result of this subtraction is (4.22). $\square$

Example 17. *Consider the $(7, 9, 13, 0)$ unlabeled GAST ($\gamma = 5$) shown in Fig. 4.4(a). Since $b_{\text{et}} = b_{\text{ut}} = 2$ and $u^0 = 5$, and aided by the unlabeled GAST tree in Fig. 4.4(b), the complexity reduction (the reduction in the number of matrices to operate on) is (see (4.22)):*

$$t' - t = 1 + \frac{1}{1!} \sum_{i_1=1}^{5} (1) = 1 + 5 = 6.$$

In other words, the cardinality of the set $\mathcal{U}$ is $t' = 11$, while from Example 13, the cardinality of the set $\mathcal{W}$ (the number of distinct WCMs) is $t = 5$. Thus, the complexity reduction is over 50%.

Corollary 2. *Given the unlabeled GAST tree, the difference between the cardinalities of the sets $\mathcal{U}$ and $\mathcal{W}$ (the reduction in the number of matrices to operate on) for a u-symmetric (a, d_1, d_2, d_3) unlabeled GAST, with the parameter $b_{\text{et}} > 0$, is:*

$$t' - t = 1 + \sum_{j=1}^{b_{\text{et}}-1} \frac{1}{j!} \prod_{i=1}^{j} u^{i-1}, \tag{4.23}$$

where $t' = |\mathcal{U}|$ and $t = |\mathcal{W}|$.

Proof. We recall again that the u-symmetric configuration is a special case of the same-size-WCMs configuration. Consequently, we can use the same idea from the proof of Corollary 1 in (4.22) to reach (4.23). $\square$

Example 18. *Consider the u-symmetric $(2\gamma, 0, \gamma^2, 0)$ unlabeled GAST. From (4.14), we know that $u^{j-1} = (\gamma - (j-1))^2$. Thus, from Corollary 2, the complexity reduction (the*

Table 4.2: Reduction in the number of matrices to operate on for different types of unlabeled GASTs.

Unlabeled GAST type	Reduction in the number of matrices ($t'-t$)
General	$t'-t = 1 + \sum_{j=1}^{b_{\text{et}}-1} \frac{1}{j!} \sum_{i_1=1}^{u^0} \sum_{i_2=1}^{u^1_{i_1}} \sum_{i_3=1}^{u^2_{i_1,i_2}} \cdots \sum_{i_j=1}^{u^{j-1}_{i_1,i_2,\dots,i_{j-1}}} T_{\text{c}}\left(u^j_{i_1,i_2,\dots,i_j}\right)$
Same-size-WCMs	$t'-t = 1 + \sum_{j=1}^{b_{\text{et}}-1} \frac{1}{j!} \sum_{i_1=1}^{u^0} \sum_{i_2=1}^{u^1_{i_1}} \sum_{i_3=1}^{u^2_{i_1,i_2}} \cdots \sum_{i_j=1}^{u^{j-1}_{i_1,i_2,\dots,i_{j-1}}} (1)$
U-symmetric	$t'-t = 1 + \sum_{j=1}^{b_{\text{et}}-1} \frac{1}{j!} \prod_{i=1}^{j} u^{i-1}$
U-symmetric, $(2\gamma, 0, \gamma^2, 0)$, with $\gamma \in \{3,4\}$	$t'-t = 1 + \sum_{j=1}^{\gamma-1} \frac{1}{j!} \prod_{i=1}^{j} (\gamma-(i-1))^2$

reduction in the number of matrices to operate on) is:

$$t'-t = 1 + \sum_{j=1}^{b_{\text{et}}-1} \frac{1}{j!} \prod_{i=1}^{j} u^{i-1} = 1 + \sum_{j=1}^{\gamma-1} \frac{1}{j!} \prod_{i=1}^{j} (\gamma-(i-1))^2 . \tag{4.24}$$

For $\gamma = 3$ (corresponding to the u-symmetric $(6,0,9,0)$ unlabeled GAST), the complexity reduction is $1+\frac{1}{1!}(9)+\frac{1}{2!}(9)(4) = 28$, which is over 80% (i.e., $t' = 34$ while $t = 6$). For $\gamma = 4$ (corresponding to the u-symmetric $(8,0,16,0)$ unlabeled GAST), the complexity reduction is $1+\frac{1}{1!}(16)+\frac{1}{2!}(16)(9)+\frac{1}{3!}(16)(9)(4) = 185$, which is about 90% (i.e., $t' = 209$ while $t = 24$).

We conclude Subsection 4.3.3 with Table 4.2. Table 4.2 lists the reduction, $t'-t$, in the number of matrices to operate on for different types of unlabeled GASTs.

Remark 16. *The analysis in Section 4.3 is focusing on the case where $b_{\text{et}} > 0$ (thus, $u^0 > 0$) because if $b_{\text{et}} = 0$ (i.e., $u^0 = 0$), $t = 1$ always. In other words, there exists only one matrix $\mathbf{W}^{\text{z}}$. As a result, there exists only one WCM of size $(\ell - d_1) \times a$, which is the single matrix $\mathbf{W}^{\text{z}}$ itself. Note that if $b_{\text{et}} > 0$, the matrix $\mathbf{W}^{\text{z}}$ of size $(\ell - d_1) \times a$ cannot be a WCM (this is the reason why we do not add 1 in (4.2) as we do in (4.19)).*

Remark 17. *An analysis similar to what we presented in Section 4.3 can be done for BASTs. However, it is omitted for brevity.*

4.4 More on How GASTs Are Removed

After demonstrating the complexity reduction achieved by operating only on the set of WCMs to remove a GAST, in this section, we provide more details on the removal of GASTs via their WCMs. We first investigate the dimension of the null space of a WCM. Then, we discuss the best that can be done to break the weight conditions of a short WCM. Finally, we discuss the exact minimum number of edge weight changes needed by the WCM framework to remove a GAST from the Tanner graph of an NB-LDPC code, and we provide a useful topological upper bound on that minimum.

4.4.1 The Dimension of the Null Space of a WCM

A GAST is removed via breaking the weight conditions of all its WCMs, i.e., via satisfying (3.8) for all its WCMs. This breaking is performed by forcing the null spaces of these WCMs to have a particular property. Thus, studying the dimension of the null space of a WCM is critical to understand how GASTs are removed.

Consider a WCM $\mathbf{W}_h^{\text{cm}}$, $1 \leqslant h \leqslant t$, of a GAST. Recall that $\mathcal{N}(\mathbf{M})$ is the null space of a matrix $\mathbf{M}$, and let $\dim(\mathcal{N}(\mathbf{M}))$ denote the dimension of the null space of a matrix $\mathbf{M}$. Moreover, let G_h^{cm} be the subgraph created by removing b_h^{cm} degree $\leqslant 2$ CNs from the GAST subgraph. The b_h^{cm} rows that are removed from $\mathbf{A}$ to reach $\mathbf{W}_h^{\text{cm}}$ correspond to these b_h^{cm} CNs. Note that these CNs are the ones on the path from the root node until the tree node linked to $\mathbf{W}_h^{\text{cm}}$ in the unlabeled GAST tree (the CNs marked as unsatisfied by Algorithm 2 to extract $\mathbf{W}_h^{\text{cm}}$). Moreover, let $\mathbf{M}(G)$ denote the adjacency matrix of a graph G.

Theorem 7. *The dimension p_h of the null space of a WCM $\mathbf{W}_h^{\text{cm}}$, $1 \leqslant h \leqslant t$, of an (a, b, d_1, d_2, d_3) GAST, given that this WCM has unbroken weight conditions, is given by:*

$$p_h = \dim\left(\mathcal{N}(\mathbf{W}_h^{\text{cm}})\right) = \sum_{k=1}^{\delta_h} \dim\left(\mathcal{N}\left(\mathbf{M}(G_{h,k}^{\text{disc}})\right)\right) \geqslant \delta_h, \tag{4.25}$$

where δ_h is the number of disconnected components in G_h^{cm}, and $G_{h,k}^{\text{disc}}$ is the kth disconnected component in G_h^{cm}, with $1 \leqslant k \leqslant \delta_h$.

Proof. It is known from graph theory that if graph G_h^{cm} has δ_h disconnected components defined as $G_{h,k}^{\text{disc}}$, $1 \leqslant k \leqslant \delta_h$, then:

$$p_h = \dim\left(\mathcal{N}(\mathbf{W}_h^{\text{cm}})\right) = \dim\left(\mathcal{N}\left(\mathbf{M}(G_h^{\text{cm}})\right)\right) = \sum_{k=1}^{\delta_h} \dim\left(\mathcal{N}\left(\mathbf{M}(G_{h,k}^{\text{disc}})\right)\right). \tag{4.26}$$

Note that from the definition of G_h^{cm}, $\mathbf{W}_h^{\text{cm}} = \mathbf{M}(G_h^{\text{cm}})$.

Then, we prove the inequality $p_h \geqslant \delta_h$. If $\exists\ G_{h,k}^{\text{disc}}$ s.t. $\dim\left(\mathcal{N}\left(\mathbf{M}(G_{h,k}^{\text{disc}})\right)\right) = 0$ (which means $\mathcal{N}\left(\mathbf{W}_h^{\text{cm}}\right) = \{\mathbf{0}\}$), then it is impossible to have a vector $\mathbf{v} = [v_1\ v_2\ \dots\ v_a]^{\text{T}} \in \mathcal{N}\left(\mathbf{M}(G_h^{\text{cm}})\right) = \mathcal{N}(\mathbf{W}_h^{\text{cm}})$ s.t. $v_f \neq 0$, $\forall f \in \{1, 2, \dots, a\}$, where a is the size of the GAST. Thus, in order to have a WCM that has unbroken weight conditions, we must have $\dim\left(\mathcal{N}\left(\mathbf{M}(G_{h,k}^{\text{disc}})\right)\right) > 0$, $\forall k \in \{1, 2, \dots, \delta_h\}$. Noting that if $\dim\left(\mathcal{N}\left(\mathbf{M}(G_{h,k}^{\text{disc}})\right)\right) > 0$, $\forall k \in \{1, 2, \dots, \delta_h\}$, then it has to be the case that $p_h = \sum_{k=1}^{\delta_h} \dim\left(\mathcal{N}\left(\mathbf{M}(G_{h,k}^{\text{disc}})\right)\right) \geqslant \delta_h$ completes the proof of Theorem 7. □

Remark 18. *Consider a WCM $\mathbf{W}_h^{\text{cm}}$ that has unbroken weight conditions. In the majority of the GASTs we have studied, if G_h^{cm} (the graph corresponding to $\mathbf{W}_h^{\text{cm}}$) has $\delta_h = 1$ (the graph is fully connected), then $\dim\left(\mathcal{N}\left(\mathbf{W}_h^{\text{cm}}\right)\right) = 1$. Similarly, we have typically observed that if $\delta_h > 1$, then $\forall G_{h,k}^{\text{disc}}$, $1 \leqslant k \leqslant \delta_h$, $\dim\left(\mathcal{N}\left(\mathbf{M}(G_{h,k}^{\text{disc}})\right)\right) = 1$. In other words, in most of the cases we have seen, $p_h = \dim\left(\mathcal{N}(\mathbf{W}_h^{\text{cm}})\right) = \delta_h$. Having said that, we have already encountered few examples where $p_h = \dim\left(\mathcal{N}(\mathbf{W}_h^{\text{cm}})\right) > \delta_h$ (see the next subsection).*

Typically, if $\delta_h = 1$, breaking the weight conditions of a WCM $\mathbf{W}_h^{\text{cm}}$ yields $\dim\left(\mathcal{N}\left(\mathbf{W}_h^{\text{cm}}\right)\right) = 0$ (there are few exceptions to that). Contrarily, it is important to note that if $\delta_h > 1$ (which again means the graph corresponding to $\mathbf{W}_h^{\text{cm}}$ has more than one disconnected components), the weight conditions of this WCM can be broken while $\dim\left(\mathcal{N}\left(\mathbf{W}_h^{\text{cm}}\right)\right) > 0$. This situation occurs if $\exists\ G_{h,k_1}^{\text{disc}}$ and G_{h,k_2}^{disc}, $1 \leqslant k_1, k_2 \leqslant \delta_h$, s.t.

$\dim\left(\mathcal{N}\left(\mathbf{M}(G^{\text{disc}}_{h,k_1})\right)\right) = 0$ while $\dim\left(\mathcal{N}\left(\mathbf{M}(G^{\text{disc}}_{h,k_2})\right)\right) > 0$. Thus, breaking the weight conditions of such a WCM by making $\dim\left(\mathcal{N}\left(\mathbf{W}^{\text{cm}}_h\right)\right) = 0$ (if possible) is, albeit sufficient, not necessary. A conceptually-similar observation will be presented in the next subsection. We present Example 19 to illustrate Theorem 7 as well as this discussion.

Example 19. *Consider the $(6,0,0,9,0)$ GAST ($\gamma = 3$) over GF(4), where GF(4) $= \{0,1,\alpha,\alpha^2\}$ and α is a primitive element (a root of the primitive polynomial $\rho(x) = x^2+x+1$, i.e., $\alpha^2 = \alpha+1$), that is shown in Fig. 4.7(a). The matrix $\mathbf{A}$ of this configuration is:*

$$\mathbf{A} = \begin{array}{c} \\ c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \\ c_8 \\ c_9 \end{array} \begin{array}{c} \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array} \\ \begin{bmatrix} w_{1,1} & \alpha & 0 & 0 & 0 & 0 \\ 0 & \alpha^2 & \alpha^2 & 0 & 0 & 0 \\ 0 & 0 & 1 & \alpha^2 & 0 & 0 \\ 0 & 0 & 0 & \alpha^2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ w_{6,1} & 0 & 0 & 0 & 0 & \alpha \\ 0 & \alpha & 0 & 1 & 0 & 0 \\ \alpha & 0 & 0 & 0 & \alpha^2 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \end{array}.$$

For the original configuration, we assume that $w_{1,1} = w_{6,1} = 1$. The unlabeled GAST tree of this configuration reveals that it is neither u-symmetric nor same-size-WCMs. The configuration has 10 WCMs (of different sizes), extracted by removing the rows of the following groups of CNs from $\mathbf{A}$: $\{(c_1,c_3,c_5),(c_1,c_4,c_9),(c_2,c_4,c_6),(c_2,c_5),(c_2,c_8),(c_3,c_6),(c_3,c_8),(c_5,c_7),(c_6,c_7),(c_7,c_8,c_9)\}$. We index these groups of CNs (and consequently, the resulting WCMs) by h, $1 \leqslant h \leqslant t = 10$. The WCM of interest in this example is $\mathbf{W}^{\text{cm}}_2$, which is extracted by removing the rows of (c_1,c_4,c_9) from $\mathbf{A}$. The graph corresponding to $\mathbf{W}^{\text{cm}}_2$, which is G^{cm}_2, is shown in Fig. 4.7(b). Note that this graph has $\delta_2 = 2$ disconnected components. For the given edge weight assignment, $\mathbf{W}^{\text{cm}}_2$ (as well as all the remaining 9 WCMs) has unbroken weight conditions. Thus, according to Theorem 7, $\dim(\mathcal{N}(\mathbf{W}^{\text{cm}}_2)) = \sum_{k=1}^{2} \dim\left(\mathcal{N}\left(\mathbf{M}(G^{\text{disc}}_{2,k})\right)\right) \geqslant 2$ must be satisfied. Solving for the null

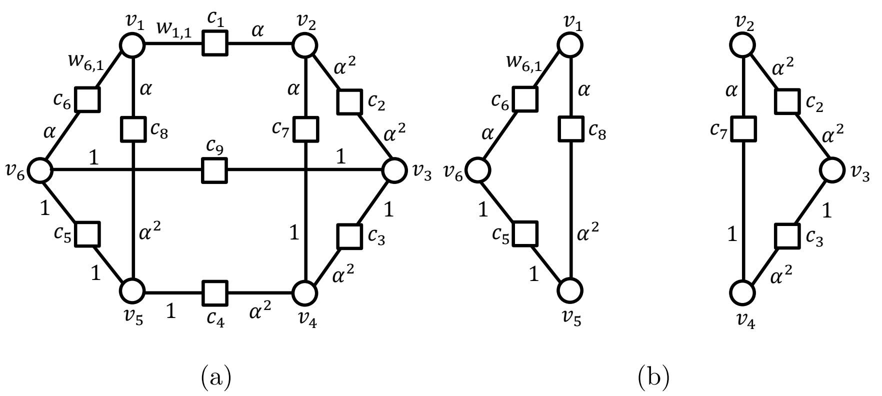


Figure 4.7: (a) A $(6, 0, 0, 9, 0)$ GAST for $\gamma = 3$. (b) The graph created by removing (c_1, c_4, c_9) from the GAST graph. GF(4) is assumed.

space of $\mathbf{W}_2^{\text{cm}}$ *yields:*

$$\mathcal{N}(\mathbf{W}_2^{\text{cm}}) = \text{span}\{[\alpha\ 0\ 0\ 0\ 1\ 1]^{\text{T}}, [0\ 1\ 1\ \alpha\ 0\ 0]^{\text{T}}\}, \tag{4.27}$$

which means that $\dim(\mathcal{N}(\mathbf{W}_2^{\text{cm}})) = 2$*, and the reason is that* $\dim\left(\mathcal{N}\left(\mathbf{M}(G_{2,k}^{\text{disc}})\right)\right) = 1$, $\forall k \in \{1, 2\}$*. Note that* $\mathcal{N}\left(\mathbf{M}(G_{2,1}^{\text{disc}})\right) = \text{span}\{[\alpha\ 1\ 1]^{\text{T}}\}$*, where* $G_{2,1}^{\text{disc}}$ *is the subgraph grouping* $\{v_1, v_5, v_6\}$ *in Fig. 4.7(b), while* $\mathcal{N}\left(\mathbf{M}(G_{2,1}^{\text{disc}})\right) = \text{span}\{[1\ 1\ \alpha]^{\text{T}}\}$*, where* $G_{2,2}^{\text{disc}}$ *is the subgraph grouping* $\{v_2, v_3, v_4\}$ *in Fig. 4.7(b). Observe that the existance of the vector:*

$$\mathbf{v} = [\alpha\ 0\ 0\ 0\ 1\ 1]^{\text{T}} + [0\ 1\ 1\ \alpha\ 0\ 0]^{\text{T}} = [\alpha\ 1\ 1\ \alpha\ 1\ 1]^{\text{T}} \in \mathcal{N}(\mathbf{W}_2^{\text{cm}}) \tag{4.28}$$

verifies that the weight conditions of $\mathbf{W}_2^{\text{cm}}$ *are unbroken.*

Now, assume that in the process of removing the GAST, we break the weight conditions of $\mathbf{W}_2^{\text{cm}}$ *via the following set of a single edge weight change:* $\{w_{6,1} : 1 \to \alpha^2\}$*. This change results in breaking the weight conditions of* $\mathbf{W}_2^{\text{cm}}$*, i.e.,* $\nexists\ \mathbf{v} = [v_1\ v_2\ \dots\ v_6] \in \mathcal{N}(\mathbf{W}_2^{\text{cm}})$ *s.t.* $v_f \neq 0$, $\forall f \in \{1, 2, \dots, 6\}$*. However,* $\mathcal{N}(\mathbf{W}_2^{\text{cm}}) = \text{span}\{[0\ 1\ 1\ \alpha\ 0\ 0]^{\text{T}}\} \neq \{\mathbf{0}\}$*, i.e.,* $\dim(\mathcal{N}(\mathbf{W}_2^{\text{cm}})) = 1$ *(it was originally* 2*). This is an example of how the weight conditions of a WCM that has a corresponding graph with* $\delta_h > 1$ *can be broken while* $\dim(\mathcal{N}(\mathbf{W}_h^{\text{cm}})) > 0$ *(for* $h = 2$ *here). Obviously, it is possible to make another edge weight change for an edge* *in $G_{2,2}^{\text{disc}}$ to make $\mathcal{N}(\mathbf{W}_2^{\text{cm}}) = \{\mathbf{0}\}$. However, this is by no means necessary for the GAST removal process.*

4.4.2 Breaking the Weight Conditions of Short WCMs

In this subsection, we discuss the best that can be done to break the weight conditions of a short WCM. The following lemma states this result.

Lemma 9. *The null space of a short WCM $\mathbf{W}_h^{\text{cm}}$ (a WCM that has fewer rows than columns) after a successful removal process for the (a, b, d_1, d_2, d_3) GAST to which this WCM belongs satisfies the following two conditions*[4]*:*

1. *Its dimension is strictly more than 0, i.e., $p_h = \dim(\mathcal{N}(\mathbf{W}_h^{\text{cm}})) > 0$.*
2. *For any $\mathbf{v} = [v_1\ v_2\ \dots\ v_a]^{\text{T}} \in \mathcal{N}(\mathbf{W}_h^{\text{cm}})$, where a is the size of the GAST, the number of non-zero elements in $\mathbf{v}$ is strictly less than a, i.e., $\|\mathbf{v}\|_0 < a$.*

Proof. Since the number of rows is less than the number of columns in this WCM, the WCM cannot have a full column rank. Thus, $\mathcal{N}(\mathbf{W}_h^{\text{cm}}) \neq \{\mathbf{0}\}$, which means $\dim(\mathcal{N}(\mathbf{W}_h^{\text{cm}})) > 0$, and proves the first condition in the lemma. Moreover, if a GAST is removed successfully, this implies that each WCM in the set $\mathcal{W}$ associated with that GAST has broken weight conditions. Thus, the second condition in the lemma is automatically satisfied for the short WCM as well. $\square$

Lemma 9 further emphasizes on the fact that the weight conditions of a WCM $\mathbf{W}_h^{\text{cm}}$ can be broken while $\dim(\mathcal{N}(\mathbf{W}_h^{\text{cm}})) > 0$ (i.e., $\mathcal{N}(\mathbf{W}_h^{\text{cm}}) \neq \{\mathbf{0}\}$). One way this case can happen is if $\delta_h > 1$ (which is discussed in the previous subsection). Another way is if the WCM is short, even with $\delta_h = 1$. For many short WCMs, the reason why this case occurs is that before breaking the weight conditions of a short WCM, it typically has $p_h = \dim(\mathcal{N}(\mathbf{W}_h^{\text{cm}})) > \delta_h$. Here, we are more interested in short WCMs with $\delta_h = 1$. The difference between the

[4]Note that Lemma 9 also applies to any WCM $\mathbf{W}_h^{\text{cm}}$ that has G_h^{cm} with $\delta_h > 1$ and at least one of the disconnected components having a short adjacency matrix (a very rare case).

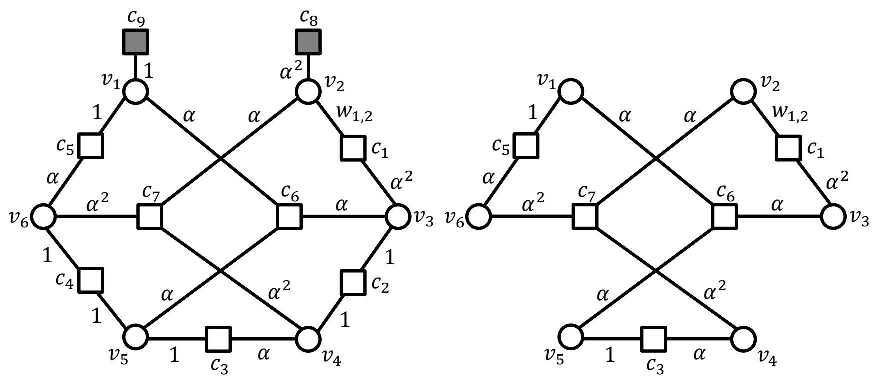
two ways is that if $\delta_h > 1$ and there does not exist any disconnected component having a short adjacency matrix, we can still break the weight conditions of the WCM by making $\dim\left(\mathcal{N}\left(\mathbf{W}_h^{\text{cm}}\right)\right) = 0$. However, such processing is not necessary, and it would require more edge weight changes than the minimum needed. Contrarily, if the WCM is short, it is impossible to break the weight conditions by making $\dim\left(\mathcal{N}\left(\mathbf{W}_h^{\text{cm}}\right)\right) = 0$, and the best we can do is what is described in Lemma 9. The following example demonstrates Lemma 9.

Example 20. *Consider the* $(6,2,2,5,2)$ *GAST (*$\gamma = 3$*) over GF(*4*), where GF(*4*)* $=$ $\{0, 1, \alpha, \alpha^2\}$ *and* α *is a primitive element, that is shown in Fig. 4.8(a). The matrix* $\mathbf{A}$ *of this configuration is:*

$$\mathbf{A} = \begin{array}{c} \\ c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \\ c_8 \\ c_9 \end{array} \begin{array}{c} \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array} \\ \left[\begin{array}{cccccc} 0 & w_{1,2} & \alpha^2 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & \alpha & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & \alpha \\ \alpha & 0 & \alpha & 0 & \alpha & 0 \\ 0 & \alpha & 0 & \alpha^2 & 0 & \alpha^2 \\ 0 & \alpha^2 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array}.$$

For the original configuration, we assume that $w_{1,2} = \alpha^2$*. From Example 12, this configuration has* 2 *WCMs, extracted by removing the rows of the following groups of CNs from* $\mathbf{A}$*:* $\{(c_3, \mathcal{O}_{\text{sg}}), (c_2, c_4, \mathcal{O}_{\text{sg}})\}$*, where* $\mathcal{O}_{\text{sg}}$ *is* (c_8, c_9)*. We index these groups of CNs (and consequently, the resulting WCMs) by* h*,* $1 \leqslant h \leqslant t = 2$*. The WCM of interest in this example is* $\mathbf{W}_2^{\text{cm}}$*, which is extracted by removing the rows of* (c_2, c_4, c_8, c_9) *from* $\mathbf{A}$*. The graph corresponding to* $\mathbf{W}_2^{\text{cm}}$*, which is* G_2^{cm}*, is shown in Fig. 4.8(b). Note that* $\mathbf{W}_2^{\text{cm}}$ *is of size* 5×6 *(a short matrix). For the given edge weight assignment,* $\mathbf{W}_2^{\text{cm}}$ *(as well as* $\mathbf{W}_1^{\text{cm}}$*) has unbroken weight conditions. Solving for the null space of* $\mathbf{W}_2^{\text{cm}}$ *yields the following:*

$$\mathcal{N}(\mathbf{W}_2^{\text{cm}}) = \text{span}\{[0\ 1\ 1\ \alpha^2\ 1\ 0]^{\text{T}}, [1\ 1\ 1\ 0\ 0\ \alpha^2]^{\text{T}}\}, \tag{4.29}$$



(a) (b)

Figure 4.8: (a) A $(6, 2, 2, 5, 2)$ GAST for $\gamma = 3$. (b) The subgraph created by removing (c_2, c_4, c_8, c_9) from the GAST subgraph. GF(4) is assumed.

This is one of the cases where we have $\dim(\mathcal{N}(\mathbf{W}_h^{\text{cm}})) > 1$ *(for* $h = 2$*) with* $\delta_h = 1$ *(the corresponding graph to* $\mathbf{W}_2^{\text{cm}}$ *is fully connected). Observe also that the existance of the vector:*

$$\mathbf{v} = [0\ 1\ 1\ \alpha^2\ 1\ 0]^{\text{T}} + \alpha[1\ 1\ 1\ 0\ 0\ \alpha^2]^{\text{T}} = [\alpha\ \alpha^2\ \alpha^2\ \alpha^2\ 1\ 1]^{\text{T}} \in \mathcal{N}(\mathbf{W}_2^{\text{cm}}) \tag{4.30}$$

verifies that the weight conditions of $\mathbf{W}_2^{\text{cm}}$ *are unbroken.*

Now, assume that in the process of removing the GAST, we break the weight conditions of $\mathbf{W}_2^{\text{cm}}$ *via the following set of a single edge weight change:* $\{w_{1,2} : \alpha^2 \to \alpha\}$. *This change results in breaking the weight conditions of* $\mathbf{W}_2^{\text{cm}}$*, i.e.,* $\nexists\, \mathbf{v} = [v_1\ v_2\ \dots\ v_6] \in \mathcal{N}(\mathbf{W}_2^{\text{cm}})$ *s.t.* $v_f \neq 0$, $\forall f \in \{1, 2, \dots, 6\}$. *However,* $\mathcal{N}(\mathbf{W}_2^{\text{cm}}) = \text{span}\{[1\ 0\ 0\ \alpha^2\ 1\ \alpha^2]^{\text{T}}\} \neq \{\mathbf{0}\}$*, i.e.,* $\dim(\mathcal{N}(\mathbf{W}_2^{\text{cm}})) = 1$ *(it was originally* 2*). This example illustrates that the weight conditions of the short WCM can only be broken with* $\dim(\mathcal{N}(\mathbf{W}_2^{\text{cm}})) > 0$ *regardless of the edge weight change(s) we perform.*

4.4.3 The Number of Edge Weight Changes Needed

In this subsection, we discuss the minimum number of edge weight changes needed, in addition to how to select these edge weight changes in order to have a successful removal of the problematic object. Recall that we need to break the weight conditions of all the WCMs of a GAST in order to remove the GAST.

Lemma 10. *The minimum number of edge weight changes (with respect to the original configuration) needed to remove an $(a, b, b_2, d_1, d_2, d_3)$ GAS (convert it into a non-AS, i.e., a non-GAS) is given by:*

$$E_{\text{GAS,min}} = g - b_{\text{vn,max}} + 1, \tag{4.31}$$

where $g = \left\lfloor \frac{\gamma-1}{2} \right\rfloor$, and $b_{\text{vn,max}}$ is the maximum number of existing unsatisfied CNs per VN in the GAS. A topological upper bound on that minimum is given by:

$$E_{\text{GAS,min}} \leqslant g - d_{1,\text{vn,max}} + 1, \tag{4.32}$$

*where $d_{1,\text{vn,max}}$ is the maximum number of existing degree-*1 *CNs per VN in the GAS.*

Proof. The set of GASs is simply the set of absorbing sets (ASs). Thus, the proof of (4.31) is exactly the same as the proof of $E_{\text{AS,min}}$ in Lemma 2.

Now, we prove the upper bound in (4.32). Recall that degree-1 CNs are always unsatisfied. Thus, irrespective of whether the VN that has the maximum number of existing unsatisfied CNs is the same as the VN that has the maximum number of existing degree-1 CNs or not, the following inequality is always satisfied:

$$b_{\text{vn,max}} \geqslant d_{1,\text{vn,max}}. \tag{4.33}$$

Substituting (4.33) in (4.31) gives (4.32), and completes the proof. □

While theoretically a GAST can be removed by forcing a single degree > 2 CN (if any) to be unsatisfied via a single edge weight change, this is not the strategy we follow. The reason is that the described process can result in another GAS with a degree > 2 unsatisfied CN ($b > d_1 + b_2$). Despite that GASs with $b > d_1 + b_2$ are generally less harmful than GASTs, it is not preferred to remove GASTs by converting some of them into other types of GASs. Thus, we remove a GAST by performing $E_{\text{GAST,min}} = E_{\text{GAS,min}}$ edge weight changes for edges connected to only degree-2 CNs (all the weights of edges connected to degree > 2 CNs remain untouched). In other words, (4.31) and (4.32) are applicable to both GASTs and GASs. Furthermore, $E_{\text{GAST,min}}$ we use in Algorithm 3 is given by (4.31), which is (2.21), and bounded by (4.32).

The upper bound in (4.32) is purely topological (determined from the unlabeled GAST), i.e., it does not require any knowledge of b of the GAST being processed (nor $b_{\text{vn,max}}$, consequently). The importance of this topological bound will be illustrated shortly.

A useful definition and a subsequent corollary, which are used to simplify the process of selecting $E_{\text{GAST,min}}$ edge weights to change, are proposed below.

Definition 17. *A* ***borderline VN*** *in an (a, b, d_1, d_2, d_3) GAST in a code with column weight γ is a VN that is connected to exactly $g = \left\lfloor \frac{\gamma-1}{2} \right\rfloor$ degree-1 CNs.*

Corollary 3. *An (a, b, d_1, d_2, d_3) GAST that has at least one borderline VN has $E_{\text{GAST,min}} = 1$, and the upper bound on $E_{\text{GAST,min}}$ is also 1.*

Proof. A borderline VN already has the maximum number of unsatisfied CNs a VN can have in a GAST (or in an AS in general), which is $g = \left\lfloor \frac{\gamma-1}{2} \right\rfloor$. Consequently, a GAST with at least one borderline VN has:

$$b_{\text{vn,max}} = d_{1,\text{vn,max}} = g = \left\lfloor \frac{\gamma - 1}{2} \right\rfloor. \tag{4.34}$$

Substituting (4.34) in (4.31) gives $E_{\text{GAST,min}} = 1$. Noting that $b_{\text{vn,max}} = d_{1,\text{vn,max}}$ proves that the upper bound is also 1 (by substituting in (4.32)). □

Remark 19. *Lemma 10, Corollary 3, and the discussion below give the minimum number of edge weight changes in addition to the specification of which edge weights need to be changed in order to remove a GAST. However, they do not determine what these particular changes should be, i.e., they do not specify the new values of the edge weights to be changed. Specifying the new values of the edge weights is performed by the WCM framework via checking the null* *spaces of all WCMs of the GAST being processed, and making sure that all the WCMs have broken weight conditions after the edge weight changes (see also Theorem 3, Algorithm 3, and Example 21). This justification is the reason why the word "**properly**" is used to describe the edge weight changes in this subsection.*

It can be concluded from Corollary 3 that any degree-2 unsatisfied CN connected to a borderline VN results in an object that is not a GAST. Thus, assuming that the object being processed is identified to be a GAST via the WCM framework (at least one of its WCMs has unbroken weight conditions), we select the edge weights to be changed based on the following two cases[5]:

1. If the GAST has at least one borderline VN ($E_{\text{GAST,min}} = 1$), then we **properly** change the weight of an edge connected to a degree-2 CN connected to any of the borderline VNs. If every VN in the GAST is borderline, then we change the weight of an edge connected to any degree-2 CN.

2. If the GAST does not have any borderline VNs ($E_{\text{GAST,min}} \geqslant 1$), then we determine the VN(s) that has (have) the maximum number, $d_{1,\text{vn,max}}$, of degree-1 CNs connected to it (them). Then we **properly** change the weights of a maximum of $(g - d_{1,\text{vn,max}} + 1)$ edges connected to different degree-2 CNs connected to a particular VN of those having $d_{1,\text{vn,max}}$ neighboring degree-1 CNs.

To relate the above analysis to the WCMs, recall that every CN in a GAST has a corresponding row in the matrix $\mathbf{A}$ of this GAST. The GAST is removed by breaking the weight conditions of all its WCMs. To achieve this, we operate on a set of rows in $\mathbf{A}$, that has the minimum cardinality and its rows correspond to degree-2 CNs, with the property that every WCM has at least one row in that set. Any set of $(g - d_{1,f} + 1)$ rows in $\mathbf{A}$ satisfies the stated property if they correspond to degree-2 CNs connected to the same VN,

[5]Note that these two items are for a stand-alone GAST. It happens in few cases that we need more than the minimum number of edge weight changes to remove a GAST because of previously removed GASTs that share edges with the GAST being processed (or other reasons).

v_f, $f \in \{1, 2, \dots, a\}$, where $d_{1,f}$ is the number of degree-1 CNs connected to VN v_f. The reason is that we cannot together remove $(g - d_{1,f} + 1)$ rows of degree-2 CNs connected to VN v_f from $\mathbf{A}$ to extract a WCM since the resulting matrix then will not be a valid $\mathbf{W}^{\text{z}}$. Thus, a set of $(g - d_{1,\text{vn,max}} + 1)$ rows of degree-2 CNs connected to the same VN that achieves $d_{1,\text{vn,max}}$ is indeed a set of minimum cardinality with the property that every WCM has at least one row in that set. Consequently, the topological upper bound in (4.32) provides the cardinality of that set of rows satisfying the stated property. Properly operating on a maximum of $(g - d_{1,\text{vn,max}} + 1)$ weights in these rows (only one weight per row) is what is needed to remove the GAST. Examples 21 and 22 illustrate the process performed by the WCM framework to remove a stand-alone GAST.

Remark 20. *Typically, we only need to perform* $(g - b_{\text{vn,max}} + 1) \leqslant (g - d_{1,\text{vn,max}} + 1)$ *edge weight changes to remove the GAST. When* $b_{\text{vn,max}} \neq d_{1,\text{vn,max}}$*, the number of WCMs with unbroken weight conditions becomes strictly less than* t*, and only* $(g - b_{\text{vn,max}} + 1)$ *rows are enough to establish a set of minimum cardinality with the property that every WCM with unbroken weight conditions has at least one row in that set.*

Example 21. *We again discuss the* $(6, 0, 0, 9, 0)$ *GAST* ($\gamma = 3$) *in Fig. 4.7(a), with* $w_{1,1} = w_{6,1} = 1$ *(see Example 19 for how the* 10 *WCMs of this GAST are extracted). Since* $b_{\text{vn,max}} = d_{1,\text{vn,max}} = 0$*, (4.31) gives* $E_{\text{GAST,min}} = \left\lfloor \frac{3-1}{2} \right\rfloor - 0 + 1 = 2$ *(same as the upper bound). Given that all VNs have* 0 *degree-*1 *neighboring CNs, any VN can be selected. Suppose that* v_1 *is selected. Each of the* 10 *WCMs has at least one of the two rows corresponding to* c_1 *and* c_6 *(both are connected to* v_1*) in* $\mathbf{A}$*. Thus, we consider the following two sets of edge weight changes for* $w_{1,1}$ *and* $w_{6,1}$ *(cardinality* 2*). The first set is* $\{w_{1,1} : 1 \to \alpha, w_{6,1} : 1 \to \alpha\}$*. Using this set of edge weight changes, the null spaces of the* 10 *WCMs become:*

$$\begin{aligned}
&\mathcal{N}(\mathbf{W}^{\text{cm}}_1) = \mathcal{N}(\mathbf{W}^{\text{cm}}_3) = \mathcal{N}(\mathbf{W}^{\text{cm}}_4) = \mathcal{N}(\mathbf{W}^{\text{cm}}_6) = \mathcal{N}(\mathbf{W}^{\text{cm}}_8) = \mathcal{N}(\mathbf{W}^{\text{cm}}_9) = \{\mathbf{0}\},\\
&\mathcal{N}(\mathbf{W}^{\text{cm}}_2) = \text{span}\{[0\ 1\ 1\ \alpha\ 0\ 0]^{\text{T}}\}, \ \textit{and}\\
&\mathcal{N}(\mathbf{W}^{\text{cm}}_5) = \mathcal{N}(\mathbf{W}^{\text{cm}}_7) = \mathcal{N}(\mathbf{W}^{\text{cm}}_{10}) = \text{span}\{[1\ 1\ 1\ \alpha\ 1\ 1]^{\text{T}}\}.
\end{aligned} \tag{4.35}$$

Clearly, the GAST is not removed as there are 3 *WCMs with unbroken weight conditions:* $\mathbf{W}_5^{\text{cm}}$*,* $\mathbf{W}_7^{\text{cm}}$*, and* $\mathbf{W}_{10}^{\text{cm}}$*. The second set is* $\{w_{1,1} : 1 \rightarrow \alpha, w_{6,1} : 1 \rightarrow \alpha^2\}$*. Using this set of edge weight changes, the null spaces of the* 10 *WCMs become:*

$$\begin{aligned}
&\mathcal{N}(\mathbf{W}_1^{\text{cm}}) = \mathcal{N}(\mathbf{W}_3^{\text{cm}}) = \mathcal{N}(\mathbf{W}_4^{\text{cm}}) = \mathcal{N}(\mathbf{W}_5^{\text{cm}}) = \mathcal{N}(\mathbf{W}_6^{\text{cm}}) = \mathcal{N}(\mathbf{W}_7^{\text{cm}}) \\
&\qquad = \mathcal{N}(\mathbf{W}_8^{\text{cm}}) = \mathcal{N}(\mathbf{W}_9^{\text{cm}}) = \mathcal{N}(\mathbf{W}_{10}^{\text{cm}}) = \{\mathbf{0}\} \text{ and} \\
&\mathcal{N}(\mathbf{W}_2^{\text{cm}}) = \text{span}\{[0\ 1\ 1\ \alpha\ 0\ 0]^{\text{T}}\},
\end{aligned} \tag{4.36}$$

which means that the GAST is successfully removed as the 10 *WCMs have broken weight conditions. As a result, it can be concluded that properly identifying which edge weights to change is important but not enough. Checking the null spaces of all WCMs is what determines which set of edge weight changes is sufficient for a successful GAST removal.*

Now, consider the case of $w_{1,1} = \alpha$ *and* $w_{6,1} = 1$ *for the original configuration (i.e., before any removal attempt). The configuration in this case is a* $(6, 1, 0, 9, 0)$ *GAST with* $b_{\text{vn,max}} = 1$ *and* $d_{1,\text{vn,max}} = 0$*. Thus, (4.31) gives* $E_{\text{GAST,min}} = 1$*, while the upper bound is* 2 *from (4.32). The null spaces of the* 10 *WCMs are:*

$$\begin{aligned}
&\mathcal{N}(\mathbf{W}_1^{\text{cm}}) = \text{span}\{[\alpha\ 1\ 1\ \alpha\ 1\ 1]^{\text{T}}\}, \\
&\mathcal{N}(\mathbf{W}_2^{\text{cm}}) = \text{span}\{[\alpha\ 0\ 0\ 0\ 1\ 1]^{\text{T}}, [0\ 1\ 1\ \alpha\ 0\ 0]^{\text{T}}\}, \text{ and} \\
&\mathcal{N}(\mathbf{W}_3^{\text{cm}}) = \mathcal{N}(\mathbf{W}_4^{\text{cm}}) = \mathcal{N}(\mathbf{W}_5^{\text{cm}}) = \mathcal{N}(\mathbf{W}_6^{\text{cm}}) = \mathcal{N}(\mathbf{W}_7^{\text{cm}}) \\
&\qquad = \mathcal{N}(\mathbf{W}_8^{\text{cm}}) = \mathcal{N}(\mathbf{W}_9^{\text{cm}}) = \mathcal{N}(\mathbf{W}_{10}^{\text{cm}}) = \{\mathbf{0}\}.
\end{aligned} \tag{4.37}$$

Only 2 *WCMs,* $\mathbf{W}_1^{\text{cm}}$ *and* $\mathbf{W}_2^{\text{cm}}$*, have unbroken weight conditions. Both of them share the row corresponding to* c_6*. Consequently, only one edge weight change is needed to break the weight conditions of the* 2 *WCMs and remove the object (*$E_{\text{GAST,min}}$ *is achieved). A set of a single edge weight change, e.g.,* $\{w_{6,1} : 1 \rightarrow \alpha^2\}$*, is sufficient to perform the removal (see also Remark 20).*

Example 22. *We discuss the* $(6, 2, 2, 5, 2)$ *GAST* ($\gamma = 3$) *in Fig. 4.8(a), with* $w_{1,2} = \alpha^2$ *(see Example 20 for how the* 2 *WCMs of this GAST are extracted). Since* $b_{\text{vn,max}} = d_{1,\text{vn,max}} = 1$, *(4.31) gives* $E_{\text{GAST,min}} = 1$ *(same as the upper bound). Either* v_1 *or* v_2 *can be selected as both are borderline VNs. Suppose that* v_2 *is selected. Each of the* 2 *WCMs has the row of* c_1 *(see Example 20). Applying the set of a single edge weight change,* $\{w_{1,2} : \alpha^2 \rightarrow \alpha\}$, *yields the following null spaces:*

$$\mathcal{N}(\mathbf{W}_1^{\text{cm}}) = \{\mathbf{0}\} \text{ and } \mathcal{N}(\mathbf{W}_2^{\text{cm}}) = \text{span}\{[1\ 0\ 0\ \alpha^2\ 1\ \alpha^2]^{\text{T}}\}, \tag{4.38}$$

which means that the GAST is successfully removed.

4.5 Removing Oscillating Sets to Achieve More Gain

Now that we have presented the in-depth analysis of the baseline WCM framework, we are ready to introduce an extension to the framework. In particular, in this section, we discuss a new set of detrimental objects, namely oscillating sets of type two (OSTs), that are the second-order cause of the error floor of NB-LDPC codes with even column weights over asymmetric channels. We show how to remove OSTs using the WCM framework. In the simulation results section, we will show that performing another optimization phase that addresses OSTs, after the GASTs removal phase, secures up to nearly 2.5 orders of magnitude overall performance gain in the error floor region over practical (asymmetric) Flash channels.

4.5.1 Defining OSs and OSTs

Before we introduce an oscillating set (OS), we define an oscillating VN.

Definition 18. *Consider a subgraph induced by a subset* $\mathcal{V}$ *of VNs in the Tanner graph of an NB-LDPC code. Set all the VNs in* $\mathcal{V}$ *to values* $\in$ *GF(q)*$\setminus\{0\}$ *and set all other VNs to* 0. *A VN in* $\mathcal{V}$ *is said to be an* ***oscillating VN*** *if the number of its neighboring satisfied CNs* *equals the number of its neighboring unsatisfied CNs for some set of VN values. The set of all oscillating VNs in $\mathcal{V}$ is referred to as $\mathcal{S}$.*

It is clear that for codes with fixed column weights (fixed VN degrees), there can exist an oscillating VN only under the condition that the column weight γ is even. Based on Definition 18, we define the oscillating set.

Definition 19. *Consider a subgraph induced by a subset $\mathcal{V}$ of VNs in the Tanner graph of an NB-LDPC code. Set all the VNs in $\mathcal{V}$ to values $\in$ GF(q)$\backslash\{0\}$ and set all other VNs to 0. The set $\mathcal{V}$ is said to be an $(a, b, b_2, d_1, d_2, d_3)$ **oscillating set (OS)** over GF(q) if and only if the size of $\mathcal{V}$ is a, the number of unsatisfied (resp., degree-2 unsatisfied) CNs connected to $\mathcal{V}$ is b (resp., b_2), the number of degree-1 (resp., 2 and > 2) CNs connected to $\mathcal{V}$ is d_1 (resp., d_2 and d_3), the set of oscillating VNs $\mathcal{S} \subseteq \mathcal{V}$ is not empty, and each VN (if any) in $\mathcal{V} \setminus \mathcal{S}$ is connected to strictly more satisfied than unsatisfied neighboring CNs, for some set of VN values.*

Recall from Chapter 3 that $\mathcal{O}$ (resp., $\mathcal{T}$ and $\mathcal{H}$) is the set of degree-1 (resp., 2 and > 2) CNs connected to $\mathcal{V}$.

The unlabeled OS is defined in a way similar to the unlabeled GAS except for that Condition 2 in Definition 6 is replaced by "*Each VN in $\mathcal{V}$ has a number of neighbors in $(\mathcal{T} \cup \mathcal{H})$ that is at least the same as its number of neighbors in $\mathcal{O}$.*" Moreover, Lemma 3 can be changed to suit OSs by referring to the unlabeled OS instead of the unlabeled GAS in the topological conditions, and by using the following equation instead of (3.2) in the weight conditions:

$$\left(\sum_{i=1}^{\ell-b} F\left(w_{i,j}\right)\right) \geqslant \left(\sum_{k=1}^{b} F\left(d_{k,j}\right)\right). \tag{4.39}$$

Note that the equality in (4.39) must hold for at least one VN in $\mathcal{V}$. We also define an oscillating set of type two (OST) as follows.

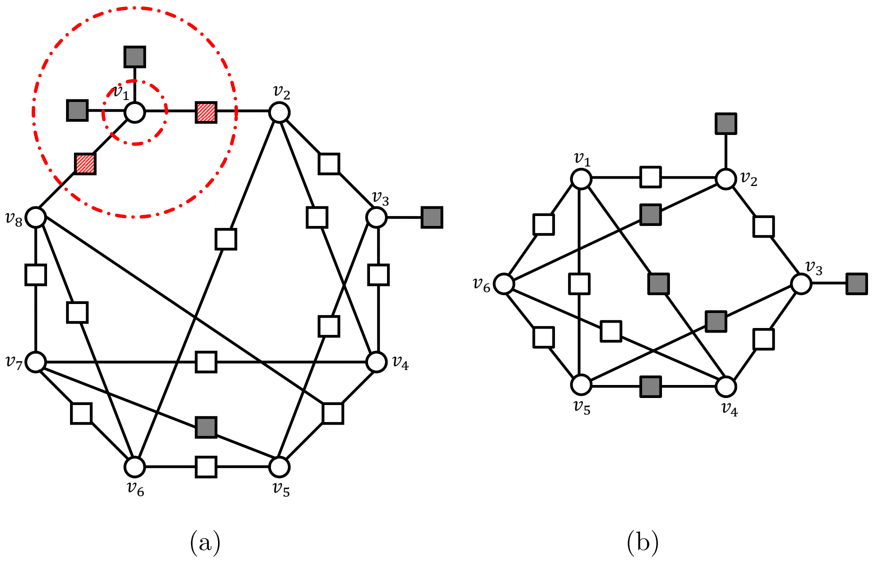


Figure 4.9: (a) An $(8, 4, 3, 13, 1)$ OST for $\gamma = 4$. (b) A $(6, 6, 2, 11, 0)$ OST for $\gamma = 4$. Appropriate non-binary edge weights are assumed. Unlabeled OSTs are reached by setting all the weights in the configurations to 1.

Definition 20. *An OS that has $d_2 > d_3$ and all the unsatisfied CNs connected to it $\in (\mathcal{O} \cup \mathcal{T})$ (having either degree 1 or degree 2), is defined as an (a, b, d_1, d_2, d_3)* ***oscillating set of type two (OST)****. In a way similar to the unlabeled OS, we also define the (a, d_1, d_2, d_3)* ***unlabeled OST****.*

If hard decision, majority rule decoding, e.g., Gallager B decoding [1], is assumed, an oscillating VN in error receives the exact same number of "stay" and "change" messages. This observation makes it harder for the decoder to correct it compared with a VN with more neighboring unsatisfied than satisfied CNs. Over aggressively asymmetric channels, oscillating VNs in error are even less likely to be corrected in many cases under soft decision decoding because of the high error magnitudes. Based on our extensive simulations, OSTs typically result in between 5% and 10% of the errors of NB-LDPC codes with even γ in the error floor region over practical (asymmetric) Flash channels, making OSTs the second-order cause, after GASTs, of the error floor in such channels. As we shall see in Section 4.6, removing OSTs from the Tanner graphs of NB-LDPC codes offers about 0.5 of an order of magnitude or more additional performance gain.

Fig. 4.9(a) shows an $(8, 4, 3, 13, 1)$ OST that has $\mathcal{S} = \{v_1\}$. Fig. 4.9(b) shows a $(6, 6, 2, 11, 0)$ OST that has $\mathcal{S} = \{v_2, v_3, v_4, v_5\}$. Some OSTs have underlying GASTs as subgraphs, while others do not. For example, if the VN v_1 is eliminated from the $(8, 4, 3, 13, 1)$ OST in Fig. 4.9(a), the underlying object is a $(7, 4, 3, 11, 1)$ GAST (the two CNs shaded in red will be degree-1 unsatisfied CNs as a result of the elimination of v_1). Contrarily, the $(6, 6, 2, 11, 0)$ OST in Fig. 4.9(b) does not have an underlying GAST.

4.5.2 How to Remove OSTs Using WCMs

Before we propose the lemma that discusses the removal of OSTs, we need to state several auxiliary results.

Lemma 11. *Consider an (a, d_1, d_2, d_3) unlabeled OST with its sets $\mathcal{T}$ and $\mathcal{H}$. A CN $c \in \mathcal{T}$ can be unsatisfied in the resulting OST (with proper edge labeling) resulting in $b > d_1$ if and only if the two neighboring VNs of c (with respect to this unlabeled OST) each has the property that strictly more than $\frac{\gamma}{2}$ of its neighboring CNs belong to $(\mathcal{T} \cup \mathcal{H})$.*

Proof. The proof follows the same logic as the proof of Theorem 1. □

Lemma 12. *Given an (a, d_1, d_2, d_3) unlabeled OST, the maximum number of unsatisfied CNs, $b_{\text{o\_max}}$, in the resulting OST after edge labeling is upper bounded by:*

$$b_{\text{o\_max}} \leqslant d_1 + b_{\text{o\_ut}}, \text{ where} \tag{4.40}$$

$$b_{\text{o\_ut}} = \left\lfloor \frac{1}{2}\left(a\left(\frac{\gamma}{2}\right) - d_1\right)\right\rfloor. \tag{4.41}$$

Proof. The proof follows the same logic as the proof of Theorem 2. The main equation in the proof is:

$$b_{\text{o\_ut}} = \sum_{f=1}^{a} \left[\frac{\gamma}{2} - b_{\text{o\_up},f}\right] = a\left(\frac{\gamma}{2}\right) - (d_1 + b_{\text{o\_ut}}), \tag{4.42}$$

where $b_{\text{o\_ut}}$ is the upper bound on the maximum number of degree-2 unsatisfied CNs the resulting OST can have after labeling, and $b_{\text{o\_up},f}$ is the number of the already-unsatisfied CNs connected to VN v_f, $f \in \{1, 2, \dots, a\}$, updated by what has been done for all the VNs processed prior to VN v_f. $\square$

The following example illustrates Lemmas 11 and 12.

Example 23. *Both configurations in Fig. 4.9 can have degree-*2 *unsatisfied CNs in the resulting OSTs. For the* $(8, 3, 13, 1)$ *unlabeled OST,* $b_{\text{o\_ut}} = 6$ *(*1 *of these* 6 *CNs is unsatisfied in the* $(8, 4, 3, 13, 1)$ *OST in Fig. 4.9(a)), while for the* $(6, 2, 11, 0)$ *unlabeled OST,* $b_{\text{o\_ut}} = 5$ *(*4 *of these* 5 *CNs are unsatisfied in the* $(6, 6, 2, 11, 0)$ *OST in Fig. 4.9(b)). The upper bound of* $b_{\text{o\_max}}$ *is achieved for both.*

For a given (a, b, d_1, d_2, d_3) OST, let $\mathcal{Z}_{\text{o}}$ be the set of all $(a, b'_{\text{o}}, d_1, d_2, d_3)$ GASTs/OSTs with $d_1 \leqslant b'_{\text{o}} \leqslant b_{\text{o\_max}}$, which have the same unlabeled GAST/OST as the original OST. Here, $b_{\text{o\_max}}$ is the largest allowable number of unsatisfied CNs for these configurations.

Definition 21. *An* (a, b, d_1, d_2, d_3) ***OST*** *is said to be* ***removed*** *from the Tanner graph of an NB-LDPC code if and only if the resulting object (after edge weight processing)* $\notin \mathcal{Z}_{\text{o}}$.

We thus augment the code optimization process for asymmetric channels to consist of two phases. The first phase, as before, focuses on the removal of GASTs, and the second phase focuses on the removal of OSTs. The ordering of the phases is critical because of the following. While it is allowed to remove a GAST by converting it to an OST during the first phase, it is not allowed to remove an OST by converting it into a GAST during the second phase because GASTs, not OSTs, are the principal cause of the error floor. This is the reason why the set $\mathcal{Z}_{\text{o}}$ is the set of all $(a, b'_{\text{o}}, d_1, d_2, d_3)$ GASTs/OSTs with $d_1 \leqslant b'_{\text{o}} \leqslant b_{\text{o\_max}}$. For example, to remove the $(6, 6, 2, 11, 0)$ OST in Fig. 4.9(b), the configuration needs to be converted into an object $\notin \{(6, 2, 2, 11, 0)$ GAST, $(6, 3, 2, 11, 0)$ GAST/OST, $(6, 4, 2, 11, 0)$ GAST/OST, $(6, 5, 2, 11, 0)$ OST, $(6, 6, 2, 11, 0)$ OST, $(6, 7, 2, 11, 0)$ OST$\}$ (this is because $d_1 = 2$ and $b_{\text{o\_max}} = d_1 + b_{\text{o\_ut}} = 7$).

For a given OST, define a matrix $\mathbf{W}^{\mathrm{z}}_{\mathrm{o}}$ to be the matrix obtained by removing b'_{o}, $d_1 \leqslant b'_{\mathrm{o}} \leqslant b_{\mathrm{o\_max}}$, rows corresponding to CNs $\in (\mathcal{O} \cup \mathcal{T})$ from the matrix $\mathbf{A}$, which is the OST adjacency matrix. These b'_{o} CNs can simultaneously be unsatisfied under some edge labeling that produces a GAST/an OST which has the same unlabeled GAST/OST as the given OST. Let $\mathcal{U}_{\mathrm{o}}$ be the set of all matrices $\mathbf{W}^{\mathrm{z}}_{\mathrm{o}}$. Each element $\in \mathcal{Z}_{\mathrm{o}}$ has one or more matrices $\in \mathcal{U}_{\mathrm{o}}$.

Definition 22. *For a given (a, b, d_1, d_2, d_3) OST and its associated adjacency matrix $\mathbf{A}$ and its associated set $\mathcal{Z}_{\mathrm{o}}$, we construct a set of t_{o} matrices as follows:*

1. *Each matrix $\mathbf{W}^{\mathrm{o\_cm}}_h$, $1 \leqslant h \leqslant t_{\mathrm{o}}$, in this set is an $(\ell - b^{\mathrm{o\_cm}}_h) \times a$ submatrix, $d_1 \leqslant b^{\mathrm{o\_cm}}_h \leqslant b_{\mathrm{o\_max}}$, formed by removing **different** $b^{\mathrm{o\_cm}}_h$ rows from the $\ell \times a$ matrix $\mathbf{A}$ of the OST. These $b^{\mathrm{o\_cm}}_h$ rows to be removed correspond to CNs $\in (\mathcal{O} \cup \mathcal{T})$ that can simultaneously be unsatisfied under some edge labeling that produces a GAST/an OST which has the same unlabeled GAST/OST as the given OST.*

2. *Each matrix $\mathbf{W}^{\mathrm{z}}_{\mathrm{o}} \in \mathcal{U}_{\mathrm{o}}$, for every element $\in \mathcal{Z}_{\mathrm{o}}$, contains at least one element of the resultant set as its submatrix.*

3. *This resultant set has the **smallest cardinality**, which is t_{o}, among all the sets which satisfy Conditions 1 and 2 stated above.*

*We refer to the matrices in this set as **oscillating weight consistency matrices (OWCMs)**, and to this set itself as $\mathcal{W}_{\mathrm{o}}$.*

In a way similar to b_{et} in GASTs, we also define $b_{\mathrm{o\_et}} \leqslant b_{\mathrm{o\_ut}}$ for OSTs such that $b_{\mathrm{o\_max}} = d_1 + b_{\mathrm{o\_et}}$. The following lemma addresses the removal of OSTs via their OWCMs. In other words, the lemma shows how the WCM framework can be customized to remove OSTs.

Lemma 13. *The necessary and sufficient processing needed to remove an (a, b, d_1, d_2, d_3) OST, according to Definition 21, is to change the edge weights such that for every OWCM*

$\mathbf{W}_h^{\text{o\_cm}} \in \mathcal{W}_{\text{o}}$, *there does not exist any vector with all its entries* $\neq 0$ *in the null space of that OWCM. Mathematically,* $\forall h$*:*

$$\text{If } \mathcal{N}(\mathbf{W}_h^{\text{o\_cm}}) = \text{span}\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{p_h}\}, \text{ then } \nexists\, \mathbf{r} = [r_1 \ r_2 \ \dots \ r_{p_h}]^{\text{T}} \text{ for}$$
$$\mathbf{v} = r_1\mathbf{x}_1 + r_2\mathbf{x}_2 + \dots + r_{p_h}\mathbf{x}_{p_h} = [v_1 \ v_2 \ \dots v_a]^{\text{T}} \text{ s.t. } v_j \neq 0, \ \forall j \in \{1, 2, \dots, a\}, \tag{4.43}$$

where p_h *is the dimension of* $\mathcal{N}(\mathbf{W}_h^{\text{o\_cm}})$*. Computations are performed over GF(q).*

Proof. The proof follows the same logic as the proof of Theorem 3. □

An analysis similar to the one in Section 4.3 can be performed to compute the number of OWCMs in the set $\mathcal{W}_{\text{o}}$, with few changes (e.g., $b_{\text{o\_et}}$ should be used instead of b_{et}). Now, we propose the minimum number of edge weight changes needed to remove an OST from the Tanner graph of an NB-LDPC code with even column weight.

Corollary 4. *The minimum number of edge weight changes (with respect to the original configuration) needed to remove an* $(a, b, b_2, d_1, d_2, d_3)$ *OS (convert it into a non-OS/non-AS) is given by:*

$$E_{\text{OS,min}} = 1 \leqslant \frac{\gamma}{2} - d_{1,\text{vn,max}} + 1, \tag{4.44}$$

where $d_{1,\text{vn,max}}$ *is the maximum number of existing degree-*1 *CNs per VN in the OS.*

Proof. The proof follows the same logic as the proof of Lemma 10 (see also Lemma 2), with $\frac{\gamma}{2}$ replacing g. Note that from the definition of an OS, at least one of its VNs has exactly $\frac{\gamma}{2}$ neighboring unsatisfied CNs. Thus,

$$E_{\text{OS,min}} = \frac{\gamma}{2} - b_{\text{vn,max}} + 1 = \frac{\gamma}{2} - \frac{\gamma}{2} + 1 = 1, \tag{4.45}$$

where $b_{\text{vn,max}}$ is the maximum number of existing unsatisfied CNs per VN in the OS, which equals $\frac{\gamma}{2}$ for any OS. □

Algorithm 4 Optimizing NB-LDPC Codes with Even Column Weights

```
1: Apply Algorithm 3 to optimize the NB-LDPC code by removing the detrimental GASTs.
2: Using initial simulations and combinatorial techniques (e.g., [6]) for the output code of
   Step 1, determine the set of OSTs to be removed.
3: Apply a customized version of Algorithm 3 to further optimize the NB-LDPC code
   generated in Step 1 by removing the detrimental OSTs.
```

As in the case of GASTs, since we only change the weights of edges connected to degree-2 CNs to remove OSTs, (4.44) also holds for $E_{\text{OST,min}}$. Moreover, a **topologically-oscillating VN** in an OST in a code with even column weight γ is connected to exactly $\frac{\gamma}{2}$ degree-1 CNs. An OST with such a VN has the upper bound on $E_{\text{OST,min}}$ equal to 1.

The following simple algorithm illustrates the procedure we follow to optimize NB-LDPC codes with even column weights for usage over asymmetric channels.

A crucial check to make while removing an OST is that the edge weight changes to be performed do not undo the removal of any of the already removed GASTs nor OSTs.

4.6 Applications of the WCM Framework

In this section, we apply the WCM framework to optimize NB-LDPC codes with different structures and for various applications, demonstrating significant performance gains in the error floor region. We used a finite-precision, fast Fourier transform based q-ary sum-product algorithm (FFT-QSPA) LDPC decoder [52], which performs a maximum of 50 iterations (except for PR channel simulations), and it stops if a codeword is reached sooner.

In the following items, we provide some details about the performed simulations and about our choices for the simulated NB-LDPC codes:

- We provide simulation results over practical storage channels, which are the main focus of this work. In particular, we present results over asymmetric Flash channels (with 3 and 6 voltage reads) and a 1-D MR channel with intrinsic memory (a PR channel). As an additional example, we also present results over the AWGN channel. These results collectively demonstrate the effectiveness of the WCM framework in optimizing

NB-LDPC codes for various channels with different characteristics.

- All our codes are circulant-based codes. The unoptimized NB block codes are chosen to be protograph-based codes as in [8] and [9] (more details about the construction are provided below). The reasons are that NB protograph-based codes enable faster encoding and decoding, and they have good performance [8, 9, 40].

- We provide results for NB-LDPC codes with various column weights (particularly $\gamma \in \{3, 4, 5\}$). The justification is that we want to demonstrate that the WCM framework typically offers at least 1 order (and up to almost 2.5 orders) of magnitude performance gain in the error floor region for NB-LDPC codes with initial average, good, and very good error floor performance (average in the case of $\gamma = 3$, good in the case of $\gamma = 4$, and very good in the case of $\gamma = 5$).

- Details about the constructions and the parameters of the SC codes we simulate are provided in Subsection 4.6.4.

All the unoptimized NB-LDPC codes we are using in Subsections 4.6.1, 4.6.2, and 4.6.3 are regular non-binary protograph-based LDPC (NB-PB-LDPC) codes. These codes are constructed as follows. First, a binary protograph matrix $\mathbf{H}^{\text{p}}$ is designed. Then, (the Tanner graph of) $\mathbf{H}^{\text{p}}$ is lifted via a lifting parameter ζ to create (the Tanner graph of) the binary image of $\mathbf{H}$, which is $\mathbf{H}^{\text{b}}$. The lifting process means that every 1 in $\mathbf{H}^{\text{p}}$ is replaced by a $\zeta \times \zeta$ circulant matrix, while every 0 (if any) in $\mathbf{H}^{\text{p}}$ is replaced by a $\zeta \times \zeta$ all-zero matrix. The circulant powers are adjusted such that the unlabeled Tanner graph of the resulting code does not have cycles of length 4. Then, the 1's in $\mathbf{H}^{\text{b}}$ are replaced by non-zero values $\in$ GF(q) to generate $\mathbf{H}$. These unoptimized codes are high performance NB-PB-LDPC codes (see also [8] and [9], in addition to Chapter 3). Note that the WCM framework works for any regular, or even irregular with fixed column weight, NB-LDPC codes. Moreover, the WCM framework also works for any GF size, q, and for any code rate.

Remark 21. *While the WCM framework works for NB-LDPC codes defined over any GF size, q, the performance gains achieved are expected to be relatively smaller for higher GF sizes, e.g., $q \geqslant 32$ (over all channels). The reason is that the fraction of edge weight assignments under which a configuration is a detrimental GAST becomes smaller as q increases (see also [10]), which means NB-LDPC codes with higher GF sizes naturally have improved error floor performance. However, increasing the GF size dramatically increases the decoding complexity, as the decoding complexity either has $O(q^2)$ or $O(q \log_2(q))$ [52]. Consequently, the approach of solely increasing the GF size in order to mitigate the error floor is not advised for storage systems because of complexity constraints. Note that NB-LDPC codes with $\gamma = 2$ can only provide good error floor performance if the GF size is large. This discussion is the reason why in our simulations, we work with various column weights ($\gamma \in \{3, 4, 5\}$), but we keep the GF size relatively small ($q \in \{4, 8\}$). In summary, we provide NB-LDPC codes with superior error floor performance, achieved via the WCM framework, without a dramatic increase in the decoding complexity. An NB-LDPC decoder implementation customized for storage application, which uses a GF size $q = 8$, is provided in [75].*

In this chapter, RBER is the raw bit error rate, which is the number of raw (uncoded) data bits in error divided by the total number of raw (uncoded) data bits read [49]. UBER is the uncorrectable bit error rate, which is a metric for the fraction of bits in error out of all bits read after the error correction is applied via encoding/decoding [49]. One formulation of UBER, as recommended by industry, is the frame error rate (FER) divided by the sector size in bits.

4.6.1 Optimizing Column Weight 5 Codes

In this subsection, we use the WCM framework to optimize NB-LDPC codes with column weight 5 for the first time. Column weight 5 codes generally guarantee better performance compared with column weight 3 and 4 codes in the error floor region. We show in this subsection, that more than 1 order of magnitude performance gain is still achievable via

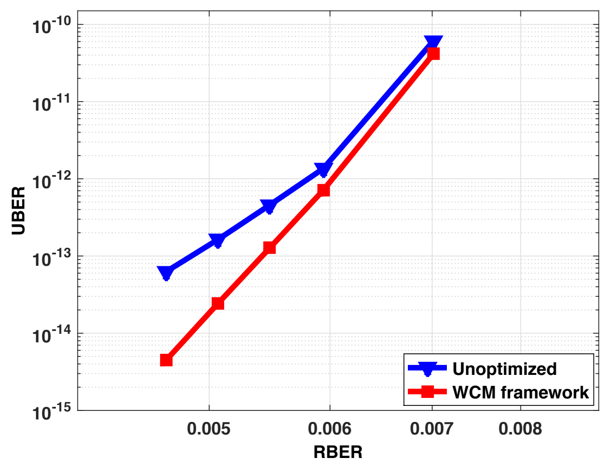


Figure 4.10: Simulation results over the NLM channel with 3 reads for Code 4.1 (unoptimized) and Code 4.2 (WCM framework). The two codes have $\gamma = 5$.

the WCM framework for such codes despite their improved error floor performance. The channel used in this subsection is a practical Flash channel: the normal-Laplace mixture (NLM) Flash channel [24]. Here, we use 3 reads, and the sector size is 512 bytes.

In the NLM channel, the threshold voltage distribution of sub-20nm multi-level cell (MLC) Flash memories is carefully modeled. The four levels are modeled as different NLM distributions, incorporating several sources of error due to wear-out effects, e.g., programming errors, thereby resulting in significant asymmetry [24]. Furthermore, the authors provided accurate fitting results of their model for program/erase (P/E) cycles up to 10 times the manufacturer's endurance specification. We implemented the NLM channel based on the parameters described in [24].

In this subsection, Code 4.1 is an NB-PB-LDPC code defined over GF(4), with block length = 6,724 bits, rate $\approx$ 0.88, and $\gamma = 5$. Code 4.2 is the result of optimizing Code 4.1 for the asymmetric NLM channel by attempting to remove the GASTs in Table 4.3 using the WCM framework.

Fig. 4.10 shows that more than 1 order of magnitude performance gain is achieved via optimizing Code 4.1 to arrive at Code 4.2 using the WCM framework. The figure also shows that using the WCM framework, an UBER of approximately 4.53×10^{-15} is achievable at

Table 4.3: Error profile of Codes 4.1 and 4.2 over the NLM channel with 3 reads, RBER $\approx 4.69 \times 10^{-3}$, UBER (unoptimized) $\approx 6.31 \times 10^{-14}$, and UBER (WCM framework) $\approx 4.53 \times 10^{-15}$ (see Fig. 4.10).

Error type	Count	
	Code 4.1	Code 4.2
$(4, 8, 8, 6, 0)$	18	0
$(6, 8, 8, 11, 0)$	9	0
$(6, 10, 8, 11, 0)$	11	0
$(7, 5, 5, 15, 0)$	4	0
$(7, 9, 9, 13, 0)$	4	0
$(7, 10, 10, 9, 2)$	7	1
$(8, 6, 6, 17, 0)$	23	0
$(8, 8, 6, 17, 0)$	15	0
Other	9	7

RBER of approximately 4.69×10^{-3} on the NLM Flash channel (an aggressively asymmetric channel) with only 3 reads.

Table 4.3 shows the error profiles of Codes 4.1 and 4.2 over the NLM channel with 3 reads. The table reveals that 33% of the errors in the error profile of Code 4.1 are non-elementary GASTs. The table also demonstrates the effectiveness of the WCM framework in removing the detrimental objects. Two of the GASTs that strongly contribute to the error profile of Code 4.1 are $(4, 8, 8, 6, 0)$ and $(8, 8, 6, 17, 0)$ GASTs, which are shown in Fig. 4.11. The key difference between GASTs in codes with $\gamma = 5$ (or 6) and GASTs in codes with $\gamma \in \{3, 4\}$ is that for the former GASTs, $g = 2$, while for the latter GASTs, $g = 1$. In other words, a VN in an object in a code with $\gamma = 5$ (or 6) can be connected to a maximum of 2 unsatisfied CNs while the object is classified as a GAST (see also Fig. 4.11 and Example 13); for $\gamma \in \{3, 4\}$, this maximum is 1.

4.6.2 Achieving More Gain by Removing Oscillating Sets

In this subsection, we demonstrate the additional gains that can be achieved for NB-LDPC codes with even column weights (particularly $\gamma = 4$) over practical asymmetric channels by removing OSTs as described in Section 4.5.

First, we present results for the NLM channel described in the previous subsection (still

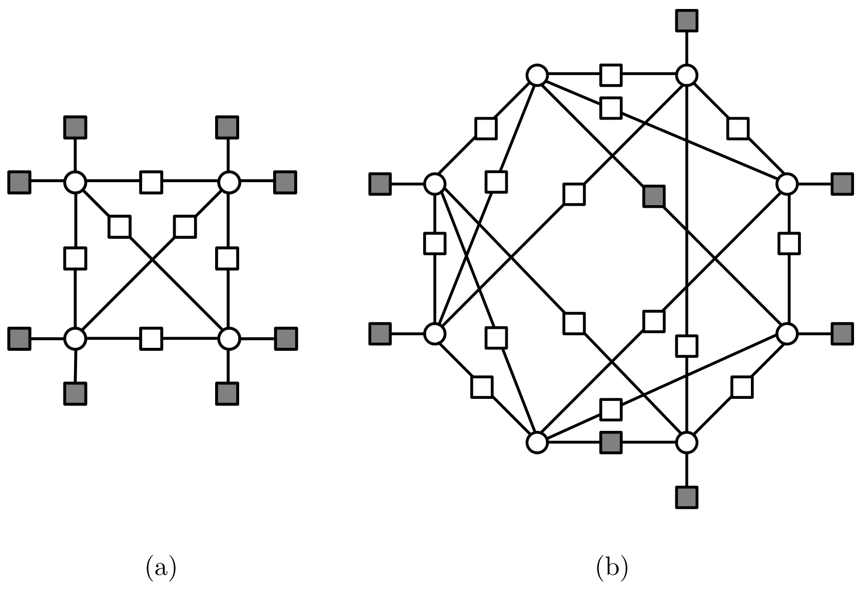


Figure 4.11: (a) A $(4, 8, 8, 6, 0)$ GAST for $\gamma = 5$. (b) An $(8, 8, 6, 17, 0)$ GAST for $\gamma = 5$. Appropriate non-binary edge weights are assumed.

with 3 reads). Code 4.3 is an NB-PB-LDPC code defined over GF(4), with block length = 8,480 bits, rate $\approx$ 0.90, and $\gamma = 4$. Code 4.4 is the result of optimizing Code 4.3 by attempting to remove the dominant GASTs $(4, 4, 4, 6, 0)$, $(6, 4, 4, 10, 0)$, $(6, 5, 5, 8, 1)$, and $(8, 4, 2, 15, 0)$ using the WCM framework (see Chapter 3). Code 4.5 is the result of optimizing Code 4.4 for the asymmetric NLM channel by attempting to remove the OSTs in Table 4.4 using the WCM framework. The performance curves of Code 4.3 (unoptimized) and Code 4.4 (WCM framework, no OSTs removal) in Fig. 4.12 were introduced in Chapter 3.

It is demonstrated by Fig. 4.12 that removing the dominant OSTs to generate Code 4.5 results in nearly 0.5 of an order of magnitude gain in performance over Code 4.4 (for which only the dominant GASTs are removed) even though Code 4.4 is highly optimized (it outperforms Code 4.3 by about 2 orders of magnitude). Thus, applying Algorithm 4 to remove OSTs after removing GASTs raises the gain to almost 2.5 orders of magnitude for Code 4.5 compared with the unoptimized code (Code 4.3) over the NLM channel. Table 4.4 shows the significant reduction in the number of OSTs in the error profile of Code 4.5 compared with Code 4.3.

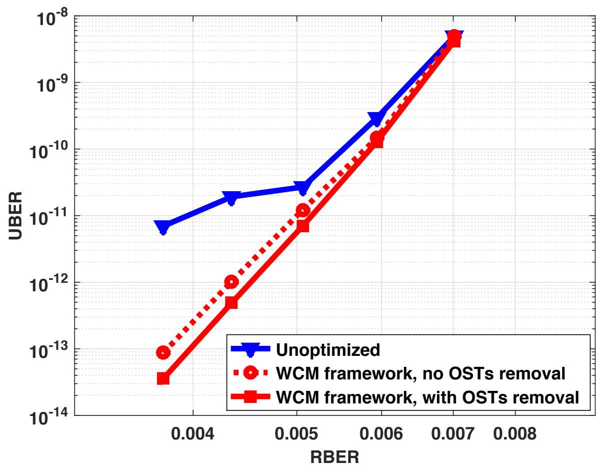


Figure 4.12: Simulation results over the NLM channel with 3 reads for Code 4.3 (unoptimized), Code 4.4 (WCM framework, no OSTs removal), and Code 4.5 (WCM framework, with OSTs removal). The three codes have $\gamma = 4$.

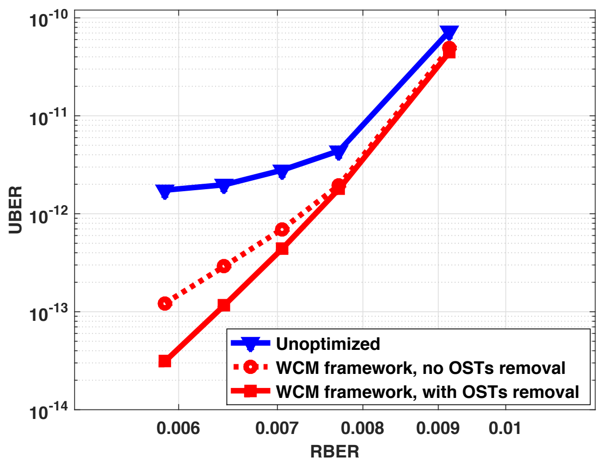


Figure 4.13: Simulation results over the CHMM channel with 3 reads for Code 4.6 (unoptimized), Code 4.7 (WCM framework, no OSTs removal), and Code 4.8 (WCM framework, with OSTs removal). The three codes have $\gamma = 4$.

Second, we present results for another asymmetric Flash channel: the Cai-Haratsch-Mutlu-Mai (CHMM) Flash channel [25]. The authors developed a model in [25] for the threshold voltage distribution that is suitable for 20nm and 24nm MLC Flash memories. The four levels are modeled as different Gaussian distributions that are shifted and broadened with the increase in P/E cycles, resulting in limited asymmetry relative to the NLM channel. We implemented the CHMM channel based on the data and the model provided in [25]. In

Table 4.4: OSTs error profile of Codes 4.3 and 4.5 over the NLM channel with 3 reads, RBER $\approx 3.75 \times 10^{-3}$, UBER (unoptimized) $\approx 6.98 \times 10^{-12}$, and UBER (WCM framework, with OSTs removal) $\approx 3.58 \times 10^{-14}$ (see Fig. 4.12).

Error type	Count	
	Code 4.3	Code 4.5
$(5, 5, 5, 6, 1)$	22	0
$(6, 5, 4, 10, 0)$	29	0
$(8, 4, 2, 12, 2)$	24	0
$(8, 5, 3, 13, 1)$	25	0

Table 4.5: OSTs error profile of Codes 4.6 and 4.8 over the CHMM channel with 3 reads, RBER $\approx 5.87 \times 10^{-3}$, UBER (unoptimized) $\approx 1.74 \times 10^{-12}$, and UBER (WCM framework, with OSTs removal) $\approx 3.11 \times 10^{-14}$ (see Fig. 4.13).

Error type	Count	
	Code 4.6	Code 4.8
$(6, 5, 2, 11, 0)$	29	0
$(6, 6, 2, 11, 0)$	11	0
$(7, 5, 3, 11, 1)$	34	0
$(8, 4, 3, 13, 1)$	15	0
$(9, 4, 2, 14, 2)$	11	1

this subsection, we use 3 reads, and the sector size is 512 bytes.

Here, Code 4.6 is an NB-PB-LDPC code defined over GF(4), with block length = 1,840 bits, rate $\approx$ 0.80, and $\gamma = 4$. Code 4.7 is the result of optimizing Code 4.6 by attempting to remove the dominant GASTs $(4, 4, 4, 6, 0)$, $(6, 4, 2, 11, 0)$, $(6, 4, 4, 10, 0)$, $(7, 4, 3, 11, 1)$, $(8, 5, 5, 12, 1)$, and $(9, 5, 5, 14, 1)$ using the WCM framework (see also Chapter 3). Code 4.8 is the result of optimizing Code 4.7 for the asymmetric CHMM channel by attempting to remove the OSTs in Table 4.5 using the WCM framework. The performance curves of Code 4.6 (unoptimized) and Code 4.7 (WCM framework, no OSTs removal) in Fig. 4.13 were introduced in Chapter 3.

Fig. 4.13 reveals that removing the dominant OSTs to design Code 4.8 results in more than 0.5 of an order of magnitude performance gain over Code 4.7 (for which only the dominant GASTs are removed). Consequently, applying Algorithm 4 to remove OSTs (after removing GASTs) raises the performance gain to more than 1.5 orders of magnitude for Code 4.8 compared with the unoptimized code (Code 4.6) over the CHMM channel. Table 4.5

clarifies the significant reduction in the number of OSTs in the error profile of Code 4.8 compared with Code 4.6.

4.6.3 Effect of Soft Information in Flash Channels

In this subsection, we show the performance of NB-LDPC codes optimized by the WCM framework over practical Flash channels with additional soft information. The NLM and CHMM Flash channels used in this subsection are as described in the previous two subsections, except that we now consider 6 voltage reads instead of 3. The additional reads increase the amount of soft information provided to the decoder from the Flash channel.

In the simulations of this subsection, Code 4.9 is an NB-PB-LDPC code defined over GF(4), with block length = 3,996 bits, rate $\approx$ 0.89, and $\gamma = 3$. Code 4.10 is the result of optimizing Code 4.9 for the asymmetric NLM channel (with 6 reads this time) by attempting to remove the dominant GASTs $(4, 2, 2, 5, 0)$, $(4, 3, 2, 5, 0)$, $(5, 2, 2, 5, 1)$, $(6, 0, 0, 9, 0)$, $(6, 1, 0, 9, 0)$, $(6, 1, 1, 7, 1)$, $(6, 2, 2, 5, 2)$, and $(6, 2, 2, 8, 0)$ using the WCM framework.

Furthermore, Code 4.11 is another NB-PB-LDPC code defined over GF(4), with block length = 3,280 bits, rate $\approx$ 0.80, and $\gamma = 4$. Code 4.12 is the result of optimizing Code 4.11 for the asymmetric NLM channel (with 6 reads) by attempting to remove the dominant GASTs $(4, 4, 4, 6, 0)$, $(6, 2, 2, 11, 0)$, $(8, 4, 3, 13, 1)$, and $(8, 5, 2, 15, 0)$ in addition to the dominant OSTs $(6, 5, 4, 10, 0)$, $(7, 6, 4, 12, 0)$, $(8, 4, 2, 12, 2)$, and $(9, 4, 2, 14, 2)$ using the WCM framework. We also reuse Code 4.6 in this subsection (its parameters are stated in the previous subsection). Code 4.13 is the result of optimizing Code 4.6 for the asymmetric CHMM channel (with 6 reads) by attempting to remove the dominant GASTs $(4, 4, 4, 6, 0)$, $(6, 4, 4, 11, 0)$, and $(7, 4, 3, 11, 1)$ in addition to the dominant OSTs $(6, 5, 2, 11, 0)$, $(7, 5, 3, 11, 1)$, $(7, 5, 4, 9, 2)$, $(7, 6, 6, 8, 2)$, $(8, 6, 2, 15, 0)$, and $(10, 7, 5, 11, 4)$ using the WCM framework.

According to our simulations, the most dominant GASTs in the error floor of the unoptimized codes (Codes 4.9, 4.11, and 4.6) are hardly affected by the additional soft information

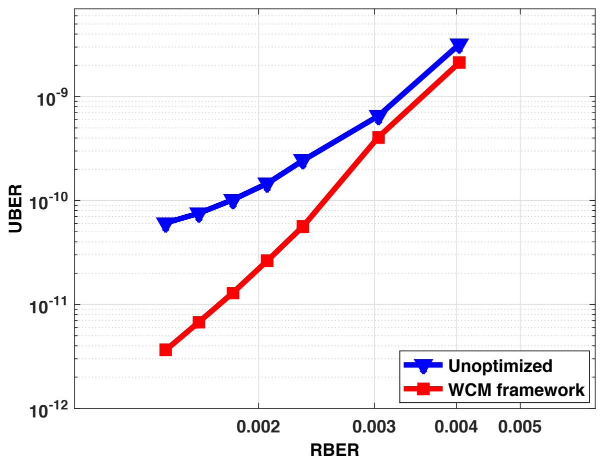


Figure 4.14: Simulation results over the NLM channel with 6 reads for Code 4.9 (unoptimized) and Code 4.10 (WCM framework). The two codes have $\gamma = 3$.

(compare the dominant GASTs listed above for Codes 4.9, 4.11, and 4.6 with the dominant GASTs in Table 3.1, Table 3.2, and Table 3.4, respectively). Moreover, Figures 4.14, 4.15, and 4.16 show that the performance gains achieved by applying the WCM framework over practical Flash channels with 6 reads are in the same range as the gains achieved over the same channels with 3 reads. In particular, more than 1 order of magnitude gain is achieved in Fig. 4.14, and more than 1.5 orders of magnitude ($>$ 0.5 of an order of magnitude is due to OSTs removal) gain is achieved in both Fig. 4.15 and 4.16. Furthermore, as in the case of 3 reads demonstrated in Chapter 3, the more asymmetric the Flash channel is, the higher the percentage of relevant non-elementary GASTs ($b > d_1$ or/and $d_3 > 0$) that appear in the error profile of the NB-LDPC code.

The major difference between the results over practical Flash channels with 3 and 6 reads is the gain achieved in RBER. Consider the $\gamma = 4$ codes simulated over the CHMM channel, and assume that the target UBER is 10^{-13}. In Fig. 4.13, Code 4.8 achieves the target UBER at RBER $\approx 6.5 \times 10^{-3}$. On the contrary, Code 4.13 achieves the target UBER at RBER $\approx 1.1 \times 10^{-2}$, as revealed by Fig. 4.16. Thus, using 6 reads achieves in this case about 70% RBER gain compared with using only 3 reads. This RBER gain is directly translated into P/E cycles gain, which means an extension in the lifetime of the Flash device. Similar gains

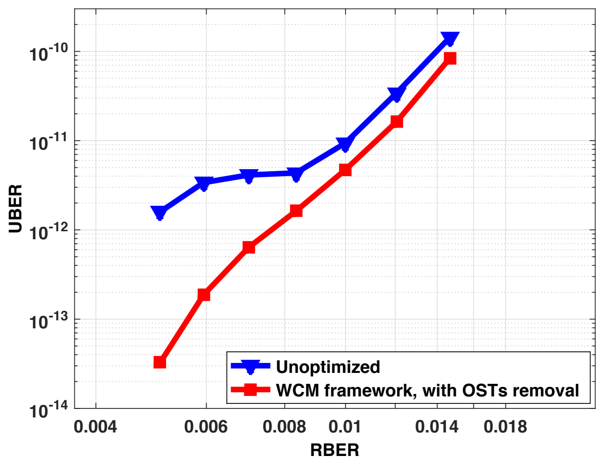


Figure 4.15: Simulation results over the NLM channel with 6 reads for Code 4.11 (unoptimized) and Code 4.12 (WCM framework, with OSTs removal). The two codes have $\gamma = 4$.

are also observed for other codes over both the NLM and CHMM channels.

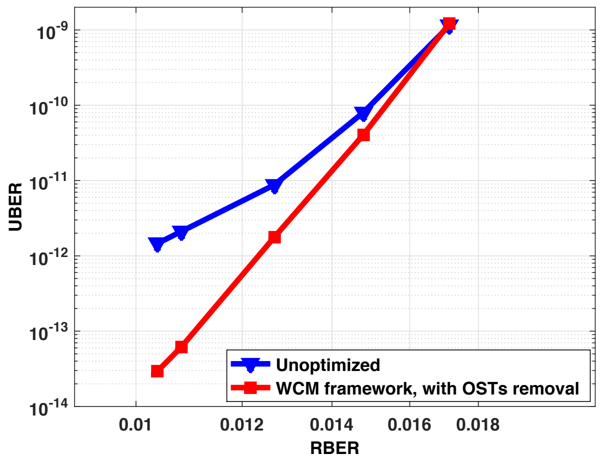


Figure 4.16: Simulation results over the CHMM channel with 6 reads for Code 4.6 (unoptimized) and Code 4.13 (WCM framework, with OSTs removal). The two codes have $\gamma = 4$.

4.6.4 Optimizing Spatially-Coupled Codes

In this subsection, we extend the scope of the WCM framework to irregular codes with fixed column weights (fixed VN degrees). In particular, we use the WCM framework to optimize non-binary spatially-coupled (NB-SC) codes with $\gamma \in \{3, 4\}$ for PR and AWGN channels, showing more than 1 order of magnitude performance gain.

SC codes are a class of graph-based (LDPC) codes that have capacity-approaching asymptotic performance, and very good finite-length performance. Literature works studying the asymptotic performance of SC codes include [31, 72, 76] for the binary case, and [73, 77] for the non-binary case. Recent results on finite-length constructions of SC codes include [30, 33, 36, 37, 74, 78, 79, 80, 81] for the binary case, and [32, 38, 44, 82] for the non-binary case. Most of these finite-length constructions are based on protographs. SC codes are constructed by partitioning an underlying block LDPC code, and then rewiring the partitioned components together multiple times [32, 37, 38, 80]. We demonstrate the effectiveness of the WCM framework by optimizing the edge weights of NB-SC codes designed using two different finite-length construction techniques (both are also based on protographs). First, we show results for NB-SC codes partitioned using single cutting vectors (CVs) [32, 80], and the underlying block LDPC codes used are array-based LDPC (AB-LDPC) codes [3]. More details about the CV technique can be found in [32]. Second, we show results for better NB-SC codes, designed using the optimal overlap, circulant power optimizer (OO-CPO) technique [37, 38, 44]. The partitioning here is derived through solving an optimization problem aiming at minimizing the total number of detrimental objects in the protograph of the SC code. Next, circulant powers of the underlying block code are optimized to further reduce the number of detrimental objects in the final unlabeled Tanner graph of the SC code. More details about the OO-CPO technique for AWGN and Flash channels can be found in [37] and [38]. In this subsection, we focus on the case of partitioning the underlying block code into only two component matrices (memory = 1), and all the SC codes do not have cycles of length 4 in their unlabeled graphs. Furthermore,

- The CV and the OO-CPO techniques mentioned above are chosen to design the underlying topologies (the binary images) of our NB-SC codes. SC codes designed using the OO-CPO technique have superior performance over AWGN [37], Flash [38], and PR channels [44].

- We use coupling lengths (see [32] and [80]) of average values (namely, 5 and 8) in our SC codes. This is because for a fixed block length, the circulant size of an SC code is inversely proportional to the coupling length. Using a very small circulant size typically exacerbates the error floor problem.

The WCM framework requires the initial unoptimized code to have a fixed column weight (fixed VN degree) but not necessarily a fixed row weight (fixed CN degree). NB-SC codes that are based on the underlying structured and regular block codes incorporate irregularities in their CN degrees (different row weights), while having fixed VN degrees [32, 37], making them suitable for optimization using the WCM framework for various applications.

We use the PR channel described in Chapter 2. This PR channel incorporates intersymbol interference (intrinsic memory), jitter, and electronic noise. The normalized channel density [50, 51, 57] is 1.4, and the PR equalization target is [8 14 2]. The receiver consists of filtering units followed by a Bahl Cocke Jelinek Raviv (BCJR) detector [53], which is based on pattern-dependent noise prediction (PDNP) [54], and an FFT-QSPA LDPC decoder [52]. The number of global (detector-decoder) iterations is 10, and the number of local (decoder only) iterations is 20. Unless a codeword is reached, the decoder performs its prescribed number of local iterations for each global iteration. More details can be found in Chapter 2.

Code 4.14 is an NB-SC code designed using the CV technique, and defined over GF(4), with block length = 8,464 bits, rate $\approx$ 0.85, and $\gamma = 3$. The underlying block code is a non-binary AB-LDPC code defined over GF(4), with circulant size = 23 and $\gamma = 3$. The coupling length $L = 8$ [32], and the underlying block code is partitioned using the optimal CV [5 11 18] (see also [32] for more details about determining the optimal CV). Code 4.15 is the result of optimizing Code 4.14 for the PR channel by attempting to remove the dominant BASTs $(6, 0, 0, 9, 0)$, $(6, 1, 0, 9, 0)$, $(6, 2, 0, 9, 0)$, $(8, 0, 0, 10, 1)$, and $(8, 0, 0, 12, 0)$ using the WCM framework.

Fig. 4.17 shows that the SC code optimized using the WCM framework (Code 4.15) outperforms the unoptimized SC code (Code 4.14) by more than 1.5 orders of magnitude over the PR channel. Note that this significant performance gain is achieved despite the

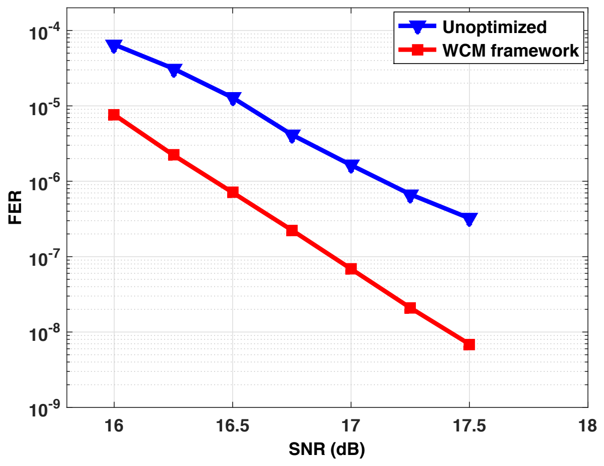


Figure 4.17: Simulation results over the PR channel for SC Code 4.14 (unoptimized) and SC Code 4.15 (WCM framework); both codes have $\gamma = 3$.

unlabeled Tanner graphs of Codes 4.14 and 4.15 both being designed using the optimal CV. In the caption of Fig. 4.17 we precede the names of Codes 4.14 and 4.15 with "SC" for clarity.

In the AWGN simulations, Code 4.16 (resp., Code 4.17) is an NB-SC code designed using the CV (resp., OO-CPO) technique, and defined over GF(8), with block length = 12,615 bits, rate ≈ 0.83, and $\gamma = 4$. The underlying block code is defined over GF(8), with circulant size = 29, $\gamma = 4$, and row weight = 29. The coupling length $L = 5$ [32, 38]. The underlying block code of Code 4.16 is partitioned using the CV [5 11 18 24], and it is a non-binary AB-LDPC code. The underlying block code of Code 4.17 is partitioned according to Fig. 4.18, upper panel, and its circulant power arrangement is given in Fig. 4.18, lower panel (see also [37] and [38]). Code 4.18 (resp., Code 4.19) is the result of optimizing Code 4.16 (resp., Code 4.17) for the AWGN channel by attempting to remove the dominant EASs $(4, 4, 4, 6, 0)$ (only from Code 4.17), $(6, 4, 4, 10, 0)$, $(6, 6, 6, 9, 0)$, and $(8, 2, 2, 15, 0)$ using the WCM framework.

Fig. 4.19 shows that the SC codes optimized using the WCM framework outperform the unoptimized SC codes by more than 1 order of magnitude over the AWGN channel. Again, note that this significant performance gain is achieved despite the unlabeled Tanner graphs of Codes 4.16 and 4.18 (resp., Codes 4.17 and 4.19) both being designed using the same technique. An important observation is that despite the very good performance of Code 4.17

●	■	●	■	●	■	●	■	●	■	●	■	●	■	●	■	●	■	●	■	●	■	●	■	●	■	●	■	■
■	●	■	●	■	●	■	●	■	●	■	●	■	●	■	●	■	●	■	●	■	●	■	●	■	●	■	●	●
●	●	●	●	●	●	●	●	●	●	●	●	●	●	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
■	■	■	■	■	■	■	■	■	■	■	■	■	■	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●

6	10	18	6	0	20	5	3	5	20	4	28	4	18	14	2	7	0	8	0	0	24	0	0	0	0	0	0	0
0	1	2	27	4	5	6	7	1	28	10	11	12	24	14	15	16	17	18	26	20	26	22	23	7	25	26	27	28
0	2	4	6	8	10	12	14	8	18	20	22	24	26	28	1	3	5	7	9	11	13	15	17	19	21	23	25	27
0	3	6	9	12	15	18	21	24	8	1	4	7	10	10	16	19	22	8	28	2	25	19	11	14	17	20	23	26

Figure 4.18: Upper panel: the OO partitioning of the underlying block code of Code 4.17 (and Code 4.19). Entries with circles (resp., squares) are assigned to the first (resp., second) component matrix. Lower panel: the circulant power arrangement for the circulants in the underlying block code of Code 4.17 (and Code 4.19) after applying the CPO.

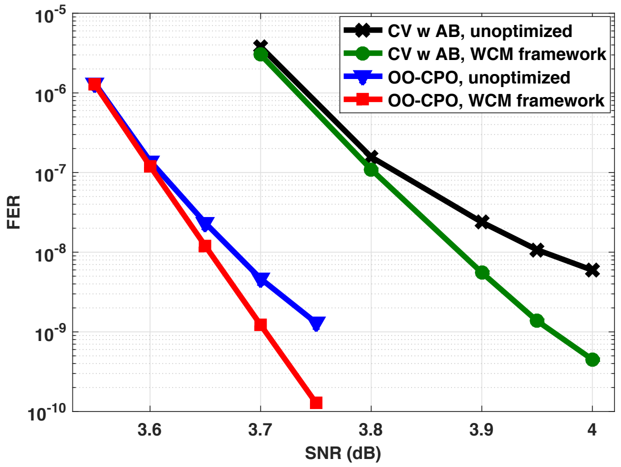


Figure 4.19: Simulation results over the AWGN channel for SC Codes 4.16 (CV, unoptimized), 4.17 (OO-CPO, unoptimized), 4.18 (CV, WCM framework), and 4.19 (OO-CPO, WCM framework); all codes have $\gamma = 4$.

(the unoptimized code is designed using the OO-CPO technique), optimizing Code 4.17 using the WCM framework to reach Code 4.19 still achieves over 1 order of magnitude performance gain. In the caption of Fig. 4.19 we precede the names of Codes 4.16, 4.17, 4.18, and 4.19 with "SC" for clarity.

4.7 Concluding Remarks

In this chapter, we have provided a theoretical analysis of a general combinatorial framework for optimizing non-binary graph-based codes. In particular, we proved the optimality of the

WCM framework, and we demonstrated its efficiency by comparing the number of matrices it operates on with that number in a suboptimal idea. We have also detailed the theory behind the removal of a GAST; we discussed the dimension of the null space of a WCM and the minimum number of edge weight changes needed to remove a GAST. Furthermore, we proposed new combinatorial objects, OSTs, and showed how to extend the WCM framework to remove them and achieve additional performance gains for NB-LDPC codes with even column weights. On the applications side, the WCM framework was applied to different codes over a variety of channels with different characteristics, where performance gains of at least 1 order, and up to nearly 2.5 orders, of magnitude were achieved. A notable extension of the WCM framework was to use it for optimizing spatially-coupled codes over multiple channels. We believe that our framework can serve as the core of an effective code optimization tool for emerging multi-dimensional ultra-dense storage devices, e.g., 3-D Flash and two-dimensional magnetic recording (TDMR) devices.

Acknowledgement

The majority of the material in this chapter is in [43], which has been recently accepted for publication. The author would like to thank the collaborators in this publication.

CHAPTER 5

High Performance Spatially-Coupled Codes

5.1 Introduction

As with other data storage systems, magnetic recording (MR) systems operate at very low frame error rate (FER) levels [11, 12, 14]. Consequently, to guarantee high error correction capability in such systems, binary [12, 14] and non-binary (NB) [83, 84, 85] graph-based codes are used. As discussed in Chapter 2, the objects that dominate the error floor region of low-density parity-check (LDPC) codes simulated in partial-response (PR) and additive white Gaussian noise (AWGN) systems are different in their combinatorial nature because of the detector-decoder looping and the intrinsic memory in PR systems. In particular, in Chapter 2, we introduced balanced absorbing sets (BASs) to characterize the detrimental objects in the case of PR (1-D MR) channels. Moreover, the weight consistency matrix (WCM) framework was introduced in Chapter 3 and Chapter 4 to systematically remove any type of absorbing sets (ASs) from the graph of an NB-LDPC code.

Spatially-coupled (SC) codes [29, 30, 31] are graph-based codes constructed by partitioning an underlying block code into components of the same size, then rewiring these components multiple times [32]. In this work, the underlying block codes, and consequently our constructed SC codes, are circulant-based (CB) codes. SC codes offer not only complexity/latency gains (if windowed decoding [33] is used), but also an additional degree of freedom in the code design; this added flexibility is achieved via partitioning of the parity check matrix of the underlying block code. This observation makes SC codes receive an increasing level of attention in multiple applications. Contiguous [32] and non-contiguous [34, 35, 36] partitioning schemes were introduced in the literature for various applications. Recently, we introduced the optimal overlap (OO), circulant power optimizer (CPO) approach to design SC codes with superior performance for AWGN [37] and practical asymmetric Flash [38] channels. The OO partitioning operates on the protograph to compute the optimal set of overlap parameters that characterizes the partitioning. The CPO operates on the unlabeled graph (edge weights are set to 1's) to adjust the circulant powers. The objective is to minimize the number of instances of a common substructure that exists in several detrimental objects. If the SC code is binary, the unlabeled graph is the final graph. If the SC code is non-binary, the WCM framework is used to optimize the edge weights after applying the OO-CPO approach as described in Chapter 3 and Chapter 4.

In this chapter, we propose an approach based on tools from combinatorics, optimization, and graph theory, to construct high performance time-invariant SC codes for PR channels, i.e., for MR devices. Unlike the case of AWGN and Flash channels (see [37] and [38]), the common substructure, whose number of instances we seek to minimize, in the case of PR channels can appear in different ways in the protograph of the SC code, making the optimization problem considerably more challenging. For that reason, we introduce the concept of the *pattern*, which is a configuration in the protograph that can result in instances of the common substructure in the unlabeled graph of the SC code after lifting. We derive an optimization problem, in which we express the weighted sum of the counts (numbers of instances) of all patterns in terms of the overlap parameters. Then, we compute the optimal set of overlap parameters (OO) that minimizes this sum. Moreover, we propose the necessary modifications to the CPO algorithm presented in [37] and [38] to make it suitable for the common substructure in the case of PR channels. We demonstrate the gains achieved by our OO-CPO (-WCM for NB SC codes) approach through tables and performance plots that compare our codes not only with SC codes, but also with CB block codes of the same length and rate.

The rest of the chapter is organized as follows. Section 5.2 introduces the necessary preliminaries. Different patterns of the common substructure are discussed in Section 5.3. The analysis of the optimization problem is presented in Section 5.4. The needed modifications over the baseline CPO are detailed in Section 5.5. We present our experimental results in Section 5.6. The chapter ends with concluding remarks in Section 5.7.

5.2 Preliminaries

In this section, we review the construction of SC codes and the definitions of the objects of interest. Here, each row (resp., column) in a parity-check matrix corresponds to a check node (CN) (resp., variable node (VN)) in the equivalent graph of the matrix (the graph of the code). Additionally, each non-zero entry in a parity-check matrix corresponds to an edge in the equivalent graph of the matrix.

Since the contribution of this work (the OO-CPO) is to optimize the topology of the underlying graph, we will focus on the unlabeled graphs and binary matrices. Labeled graphs and non-binary matrices will be discussed as needed. Let $\mathbf{H}$ be the binary parity-check matrix of the underlying regular CB code that has column weight (VN degree) γ and row weight (CN degree) κ. This matrix consists of $\gamma\kappa$ circulants. Each circulant is of the form $\sigma^{f_{i,j}}$, where $0 \leqslant i \leqslant \gamma - 1$, $0 \leqslant j \leqslant \kappa - 1$, and σ is the $z \times z$ identity matrix after cyclically shifting its columns one unit to the left. Circulant powers are $f_{i,j}$, $\forall i, j$, and they are defined, in addition to z, as the lifting parameters. Separable CB (SCB) codes have $f_{i,j} = f(i)f(j)$. The underlying block codes we use to design SC codes in this work are CB codes with no zero circulants and with $z > \kappa$.

The binary SC code is constructed as follows. First, $\mathbf{H}$ is partitioned into $(m+1)$ disjoint component matrices (they all have the same size as $\mathbf{H}$): $\mathbf{H}_0, \mathbf{H}_1, \dots, \mathbf{H}_m$, where m is defined as the memory of the SC code. Each component matrix $\mathbf{H}_y$, $0 \leqslant y \leqslant m$, contains some of the $\gamma\kappa$ circulants of $\mathbf{H}$ and zero circulants elsewhere such that $\mathbf{H} = \sum_{y=0}^{m} \mathbf{H}_y$. Our approach

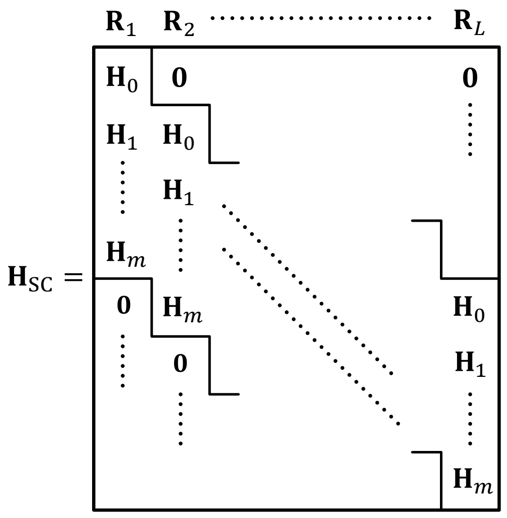


Figure 5.1: The parity-check matrix of an SC code with parameters m and L.

is general; it works for any m and any $\gamma \geqslant 3$. Then, $\mathbf{H}_0, \mathbf{H}_1, \dots, \mathbf{H}_m$ are coupled L times, as shown in Fig. 5.1, to construct the binary parity-check matrix of the SC code, $\mathbf{H}_{\mathrm{SC}}$, which is of size $\gamma z(L+m) \times \kappa z L$. A replica is any $\gamma z(L+m) \times \kappa z$ submatrix of $\mathbf{H}_{\mathrm{SC}}$ that contains $\left[\mathbf{H}_0^{\mathrm{T}}\ \mathbf{H}_1^{\mathrm{T}}\ \dots\ \mathbf{H}_m^{\mathrm{T}}\right]^{\mathrm{T}}$ and zero circulants elsewhere. Replicas are denoted by $\mathbf{R}_\rho$, $1 \leqslant \rho \leqslant L$.

The protograph matrix (PM) of a binary CB matrix is the matrix resulting from replacing each $z \times z$ non-zero circulant with 1, and each $z \times z$ zero circulant with 0. The PMs of $\mathbf{H}$ and $\mathbf{H}_y$, $0 \leqslant y \leqslant m$, are $\mathbf{H}^{\mathrm{p}}$ and $\mathbf{H}_y^{\mathrm{p}}$, respectively, and they are all of size $\gamma \times \kappa$. The PM of $\mathbf{H}_{\mathrm{SC}}$ is $\mathbf{H}_{\mathrm{SC}}^{\mathrm{p}}$, and it is of size $\gamma(L+m) \times \kappa L$. This $\mathbf{H}_{\mathrm{SC}}^{\mathrm{p}}$ also has L replicas, $\mathbf{R}_\rho$, $1 \leqslant \rho \leqslant L$, but with 1×1 circulants. Non-binary SC (NB-SC) codes can be constructed from binary SC codes as described in [38] and guided by Chapter 4. The NB codes we use in this work have parity-check matrices with their elements in GF(q), where GF refers to Galois field, $q = 2^\lambda$ is the GF size (order), and $\lambda \in \{2, 3, \dots\}$ (in the binary case, $q = 2$).

A contiguous technique for partitioning $\mathbf{H}$ to construct $\mathbf{H}_{\mathrm{SC}}$, namely cutting vector (CV) partitioning, was investigated aiming to generate SC codes for PR channels [32]. Multiple non-contiguous partitioning techniques were recently introduced in the literature, e.g., minimum overlap (MO) partitioning [34, 35], general edge spreading [36], in addition to OO partitioning [37, 38]. These non-contiguous partitioning techniques significantly outperform contiguous ones [34, 37, 38]. However, as far as we know, no prior work has proposed non-contiguous techniques in the context of PR channels. The goal of this work is to derive the effective OO-CPO approach for partitioning and lifting to construct high performance SC codes optimized for PR channels.

Consider the graph of an LDPC code. An (a, b) AS in this graph is defined as a set of a VNs with b unsatisfied CNs connected to it such that each VN is connected to strictly more satisfied than unsatisfied CNs, for some set of VN values (these a VNs have non-zero values, while the remaining VNs are set to zero) [3]. For canonical channels, e.g., the AWGN channel, elementary ASs (EASs) are the objects that dominate the error floor region of LDPC codes. EASs have the property that all satisfied CNs are of degree 2, and all unsatisfied CNs are of degree 1 [10, 86]. The different characteristics of storage channels (compared with the AWGN channel) result in changing the combinatorial properties of detrimental objects in graph-based codes simulated over such channels (see the previous chapters).

The intrinsic memory in PR channels [11] can result in VN errors having high magnitudes, which is typically not the case for canonical channels. These VN errors with high magnitudes make it very difficult for unsatisfied CNs with degree > 1 to participate in correcting an AS error. Consequently, it becomes more likely to have absorbing set errors with unsatisfied CNs having degree $\geqslant 2$, which are non-elementary absorbing set errors. Moreover, the detector-decoder looping (global iterations) help the decoder correct AS errors with higher numbers of unsatisfied CNs. Thus, the objects that dominate the error floor region of LDPC codes simulated over PR channels can be non-elementary, and they have a fewer number of unsatisfied (particularly degree-1) CNs, which is the reason why they are called "balanced". BASs and BASs of type two (BASTs) were introduced in the previous chapters (see also [40] and [42]) to capture such detrimental objects.

We now present the definitions of different objects of interest. Examples on these objects of interest are in Fig. 5.2. Let $g = \left\lfloor \frac{\gamma-1}{2} \right\rfloor$, which is the maximum number of unsatisfied CNs a VN can have in an AS.

Definition 23. *Consider a subgraph induced by a subset $\mathcal{V}$ of VNs in the (Tanner) graph of a code. Set all the VNs in $\mathcal{V}$ to values $\in$ GF(q)$\backslash\{0\}$ and set all other VNs to* 0. *The set $\mathcal{V}$ is said to be an (a, b, d_1, d_2, d_3)* ***balanced absorbing set of type two (BAST)*** *over GF(q) if the size of $\mathcal{V}$ is a, the number of unsatisfied CNs connected to $\mathcal{V}$ is b, $0 \leqslant b \leqslant \lfloor \frac{ag}{2} \rfloor$, the number of degree-*$1$ *(resp.,* 2 *and* > 2*) CNs connected to $\mathcal{V}$ is d_1 (resp., d_2 and d_3), $d_2 > d_3$, all the unsatisfied CNs connected to $\mathcal{V}$ (if any) have either degree* 1 *or degree* 2*, and each VN in $\mathcal{V}$ is connected to strictly more satisfied than unsatisfied neighboring CNs, for some set of VN values.*

While the above definition was introduced in the context of non-binary codes, as shown in Chapter 2 and Chapter 3, it is valid in the binary case as well (set $q = 2$). An (a, d_1, d_2, d_3) unlabeled BAST (UBS) is a BAST with the weights of all edges of its graph replaced by 1's. All our abbreviations are short-handed for simplicity.

Definition 24. *Let $\mathcal{V}$ be a subset of VNs in the unlabeled graph (all edge weights are* 1*'s) of a code. Let $\mathcal{O}$ (resp., $\mathcal{T}$ and $\mathcal{H}$) be the set of degree-*1 *(resp.,* 2 *and* > 2*) CNs connected to $\mathcal{V}$. This graphical configuration is an (a, d_1)* ***unlabeled elementary trapping set (UTS)*** *if $|\mathcal{V}| = a$, $|\mathcal{O}| = d_1$, and $|\mathcal{H}| = 0$. A UTS is an* ***unlabeled elementary absorbing set (UAS)*** *if each VN in $\mathcal{V}$ is connected to strictly more neighbors in $\mathcal{T}$ than in $\mathcal{O}$.*

A binary protograph configuration is also defined by (a, d_1) for simplicity. The WCM framework removes a BAST from the graph of an NB code by careful processing of its edge weights (see Chapter 3 and Chapter 4 for details).

5.3 The Common Substructure and Its Patterns

The idea of focusing on a common substructure in the design of the unlabeled graph of an SC code simplifies the optimization procedure. Additionally, minimizing the number of instances of the common substructure significantly reduces the multiplicity of several, different types of detrimental objects simultaneously [32, 37], which is a lot more feasible compared with

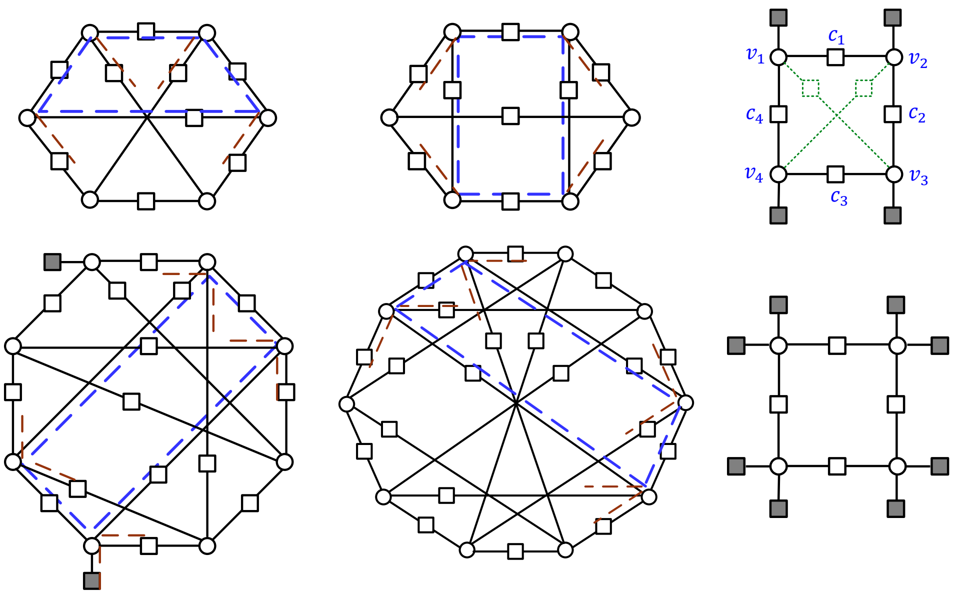


Figure 5.2: The UBSs of multiple detrimental BASTs and the associated common substructures. Upper panel ($\gamma = 3$): two non-isomorphic $(6, 0, 9, 0)$ UBSs, and the common substructure is the $(4, 4)$ UAS. Lower panel ($\gamma = 4$): an $(8, 2, 15, 0)$ UBS and a $(10, 0, 20, 0)$ UBS, and the common substructure is the $(4, 8)$ UTS. Common substructures are marked with dashed blue and dashed brown lines. Internal connections in a cycle of length 8 are shown in dotted green lines in the $(4, 4)$ UAS.

operating on all these detrimental objects separately (especially for partitioning). It was shown in [32] that the $(4, 4(\gamma-2))$ UAS/UTS, $\gamma \geqslant 3$, is the common substructure of interest for PR channels (unlike the case for AWGN [36, 37] and Flash channels [38], where the substructure of interest is the $(3, 3(\gamma-2))$). Fig. 1 shows UBSs of multiple detrimental BASTs for codes with $\gamma \in \{3, 4\}$ simulated over PR channels, demonstrating that the common substructure of interest is the $(4, 4(\gamma - 2))$ UAS/UTS.

We note that the $(4, 4(\gamma - 2))$ UAS/UTS is a cycle of length 8 ***with no internal connections*** (ignore degree-1 CNs). From [87] (see also [38]), it is known that each cycle in the unlabeled graph (the graph of $\mathbf{H}_{\text{SC}}$) is derived from a configuration in the protograph (the graph of the PM $\mathbf{H}^{\text{p}}_{\text{SC}}$) under specific conditions on the powers of the circulants involved in that cycle. Thus, in the OO stage, we operate on the protograph. Then, in the CPO stage, we operate on the circulant powers.

Remark 22. *Let x^{-a} (resp., x^{-b}) be an integer s.t. $2 \leqslant x^{-a} \leqslant x$ (resp., $0 \leqslant x^{-b} \leqslant x$). Note that a $(4^{-a}, (4(\gamma-2))^{-b})$ configuration in the protograph can result in $(4, 4(\gamma-2))$ UASs/UTSs in the unlabeled graph depending on the circulant powers. Thus, in the OO stage, we operate on all protograph configurations that can result in $(4, 4(\gamma-2))$ UASs/UTSs (cycles of length 8 with no internal connections) in the unlabeled graph, including the protograph configurations that do have internal connections. Then in the CPO stage, we treat the $(4, 4(\gamma-2))$ UASs/UTSs and the $(4, 4(\gamma-2)-2\delta)$ UASs/UTSs differently, where $\delta \in \{1, 2\}$ is the number of existing internal connections in the configuration after lifting.*

The major difference between the $(4, 4(\gamma-2))$ UAS/UTS and the $(3, 3(\gamma-2))$ UAS/UTS is that there are multiple configurations in the protograph that can generate the former object in the unlabeled graph. We call these different configurations ***patterns***. A pattern is defined by the dimensions of the matrix of its subgraph. The following lemma investigates the number and nature of these patterns.

Lemma 14. *The number of distinct patterns (with different dimensions) in the protograph of a code that can result in $(4, 4(\gamma-2))$ UASs/UTSs in the unlabeled graph of the code after lifting is 9, in the case of $\gamma \geqslant 4$. The numbers of CNs and VNs in these 9 patterns are both in $\{2, 3, 4\}$. This number of distinct patterns reduces to 7 in the case of $\gamma = 3$.*

Proof. Since the objects of interest in the unlabeled graph are cycles of length 8 with 4 CNs and 4 VNs, a protograph pattern that can generate some of them must have at most 4 CNs and 4 VNs. Moreover, to result in cycles of length 8 after lifting, the pattern must have at least 2 CNs and 2 VNs. Combining these two statements yields that the numbers of CNs and VNs of a protograph pattern that can result in $(4, 4(\gamma-2))$ UASs/UTSs in the unlabeled graph must be in $\{2, 3, 4\}$.

Consequently, in order to have 9 distinct patterns for the case of $\gamma \geqslant 4$, we show that selecting any number of CNs in $\{2, 3, 4\}$ and any number of VNs in $\{2, 3, 4\}$ can result in a distinct pattern (one or more instances) that is capable of generating cycles of length 8 in the unlabeled graph. Fig. 5.3 demonstrates the above statement, focusing on the matrix representation of patterns and cycles. In the case of $\gamma = 3$, a pattern cannot have 4 ones in a column, which reduces the number of distinct patterns to 7. $\square$

We define the 9 patterns according to the dimensions of their submatrices in $\mathbf{H}_{\text{SC}}^{\text{p}}$ as follows. Pattern P_1 is 2×2, Pattern P_2 is 2×3, Pattern P_3 is 3×2, Pattern P_4 is 2×4, Pattern P_5 is 4×2, Pattern P_6 is 3×3, Pattern P_7 is 3×4, Pattern P_8 is 4×3, and Pattern P_9 is 4×4 (all illustrated in Fig. 5.3).

Remark 23. *Following the same logic we used in Lemma 14 and its proof for the* $(3, 3(\gamma-2))$ *UAS/UTS, leads to a possibility to also have patterns for this case, with the number of CNs and VNs in* $\{2, 3\}$*. However, a careful analysis guides to the fact that only one protograph pattern can result in* $(3, 3(\gamma-2))$ *UASs/UTSs (cycles of length* 6*) after lifting, which is the* 3×3 *pattern, and it is itself a cycle of length* 6 *[37, 38].*

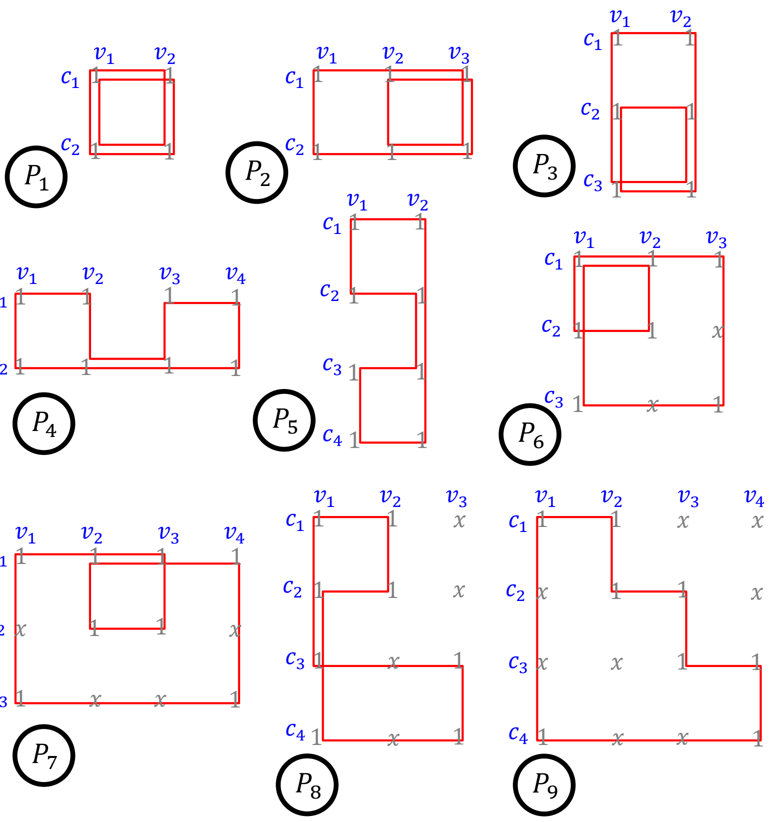


Figure 5.3: The 9 protograph patterns that can result in cycles of length 8 in the unlabeled graph after lifting. One way of traversing each pattern to generate cycles of length 8 is depicted in red. Note that only Pattern P_9 represents a cycle of length 8 in the protograph.

The following lemma discusses the relation between different protograph patterns and the resulting cycles after lifting. Define a ***cycle-*8 *candidate*** of Pattern P_ℓ as a way to traverse P_ℓ in order to reach cycles of length 8 in the unlabeled graph of the code after lifting. Some candidates are shown in Fig. 5.3.

Lemma 15. *Let ζ_{P_ℓ} be the number of distinct cycle-8 candidates of Pattern P_ℓ. Then,*

$$\zeta_{P_\ell} = \begin{cases} 1, & \ell \in \{1,6,9\}, \\ 2, & \ell \in \{7,8\}, \\ 3, & \ell \in \{2,3\}, \\ 6, & \ell \in \{4,5\}. \end{cases} \tag{5.1}$$

Proof. We define a cycle-8 candidate according to the connectivity as follows: $c_1 - v_1 - c_2 - v_2 - c_3 - v_3 - c_4 - v_4$ (each CN connects the next two VNs in a circular fashion, see Fig. 5.2). From Fig. 5.3, there is only one cycle-8 candidate for Pattern P_1, which is $c_1 - v_1 - c_2 - v_2 - c_1 - v_1 - c_2 - v_2$, and this is the case for all square patterns. Thus, $\zeta_{P_\ell} = 1$ for $\ell \in \{1,6,9\}$. It can be understood from Fig. 5.3 that $\zeta_{P_\ell} \neq 1$ for all the remaining patterns. In particular, we have two cycle-8 candidates for Pattern P_7, that are: $c_1 - v_1 - c_2 - v_2 - c_1 - v_3 - c_3 - v_4$ and $c_1 - v_1 - c_2 - v_3 - c_1 - v_2 - c_3 - v_4$ (which is the red cycle on P_7 in Fig. 5.3). The situation is the same for Pattern P_8 because it is the transpose of P_7. Thus, $\zeta_{P_\ell} = 2$ for $\ell \in \{7,8\}$. The rest of the cases can be derived similarly. □

Pattern P_1 has $\zeta_{P_\ell} = 1$ (see (5.1)), and it results in $z/2$ or 0 cycles of length 8 after lifting (since P_1 is only 2×2), while all the remaining patterns result in z or 0 cycles of length 8 after lifting [38, 87]. Thus, we define the ***pattern weight***, β_{P_ℓ}, which plays an important role in the discrete optimization problem of the OO, as follows:

$$\beta_{P_\ell} = \begin{cases} 1/2, & \ell = 1, \\ \zeta_{P_\ell}, & \ell \in \{2,3,4,5,6,7,8,9\}. \end{cases} \tag{5.2}$$

5.4 OO: Building and Solving the Optimization Problem

Now, we are ready to build the optimization problem. Consider the protograph of an SC code. The **weighted sum** of the total number of instances of all patterns is given by:

$$F_{\text{sum}} = \sum_{\ell=1}^{9} \beta_{P_\ell} F_{P_\ell}, \tag{5.3}$$

where F_{P_ℓ} is the total number of instances of Pattern P_ℓ. The goal is to express F_{sum}, through F_{P_ℓ}, $\forall \ell$, as a function of the overlap parameters, then find the optimal set of overlap parameters that minimize F_{sum} for OO partitioning. We first recall the definition and the properties of overlap parameters. More details on that part can be found in [37].

Definition 25. *For any m, let $\mathbf{\Pi}_1^1 = \left[\mathbf{H}_0^{\text{T}}\ \mathbf{H}_1^{\text{T}}\ \dots\ \mathbf{H}_m^{\text{T}}\right]^{\text{T}}$, and let $\mathbf{\Pi}_1^{1,\text{p}}$ be its PM (of size $(m+1)\gamma \times \kappa$). A **degree-$\mu$ overlap** among μ rows (or CNs) of $\mathbf{\Pi}_1^{1,\text{p}}$ indexed by $\{i_1, \dots, i_\mu\}$, $1 \leqslant \mu \leqslant \gamma$, $0 \leqslant i_1, \dots, i_\mu \leqslant (m+1)\gamma - 1$, is defined as a position (column) in which all these rows have 1's simultaneously. A **degree-μ overlap parameter**, $t_{\{i_1,\dots,i_\mu\}}$, is defined as the number of degree-μ overlaps among the rows indexed by $\{i_1, \dots, i_\mu\}$ in $\mathbf{\Pi}_1^{1,\text{p}}$. A degree-$1$ overlap parameter t_{i_1}, $0 \leqslant i_1 \leqslant (m+1)\gamma - 1$, is defined as the number of 1's in row i_1 of $\mathbf{\Pi}_1^{1,\text{p}}$.*

Note that a degree-μ overlap parameter, if $\mu > 1$, is always zero if in the set $\{i_1, \dots, i_\mu\}$ there exists at least one pair of distinct row indices, say (i_{τ_1}, i_{τ_2}), with the property that $i_{\tau_1} \equiv i_{\tau_2} \pmod{\gamma}$ [37]. Define the set of all non-zero overlap parameters as $\mathcal{O}$. The parameters in $\mathcal{O}$ are not entirely independent. The set of all independent non-zero overlap parameters, $\mathcal{O}_{\text{ind}}$, is:

$$\begin{aligned}\mathcal{O}_{\text{ind}} = \{t_{\{i_1,\dots,i_\mu\}} \mid 1 \leqslant \mu \leqslant \gamma,\ 0 \leqslant i_1, \dots, i_\mu \leqslant m\gamma - 1,\\ \forall \{i_{\tau_1}, i_{\tau_2}\} \subseteq \{i_1, \dots, i_\mu\}\ i_{\tau_1} \not\equiv i_{\tau_2} \pmod{\gamma}\}.\end{aligned} \tag{5.4}$$

The other non-zero overlap parameters in $\mathcal{O} \setminus \mathcal{O}_{\text{ind}}$ are obtained from the parameters in $\mathcal{O}_{\text{ind}}$

according to [37, Lemma 3]. The cardinality of the set $\mathcal{O}_{\text{ind}}$, which determines the complexity of the discrete optimization problem of the OO stage, is given by (see also [37, Lemma 4] for more details):

$$\mathcal{N}_{\text{ind}} = |\mathcal{O}_{\text{ind}}| = \sum_{\mu=1}^{\gamma} m^{\mu} \binom{\gamma}{\mu} = (m+1)^{\gamma} - 1. \tag{5.5}$$

As demonstrated in Fig. 5.3, for all the patterns of interest, the highest overlap degree is $\mu = 4$ (a pattern has at most 4 CNs). Note that while the overlap parameters themselves must be restricted to $\mathbf{\Pi}_1^{1,\text{p}}$, the concept of the degree-μ overlap can be generalized from $\mathbf{\Pi}_1^{1,\text{p}}$ to the PM of the SC code, $\mathbf{H}_{\text{SC}}^{\text{p}}$. We will use this generalization in the analysis of patterns.

We aim at expressing F_{P_ℓ}, $\forall \ell$, in terms of the parameters in $\mathcal{O}_{\text{ind}}$. Let $\mathbf{R}_r$ be a replica in which at least one VN of the pattern being studied exists. We call $\mathbf{R}_r$ the reference replica. Moreover, let the CNs (or rows) of the pattern be of the form $c_x = (r-1)\gamma + i_x$, $1 \leqslant x \leqslant 4$. Here, c_x is the index of the row in $\mathbf{H}_{\text{SC}}^{\text{p}}$ corresponding to the CN. In the following, we consider the protograph of an SC code with parameters $\gamma \geqslant 3$, κ, m, L, and $\mathcal{O}$. We define $(x)^+ = \max\{x, 0\}$, and $F_{P_\ell,1}^{k}$ as the number of instances of Pattern P_ℓ that start at replica $\mathbf{R}_1$ and span k consecutive replicas. Here, "start" and "span" are both with respect to the VNs of these instances. Note that each VN in a pattern corresponds to an overlap (see Fig. 5.3).

As we shall see later, a Pattern P_ℓ spans at most χ consecutive replicas, where χ either $= m+1$ or $= 2m+1$, depending on the value of ℓ. Thus, in the math, we consider the case of $L \geqslant \chi$.

We say here that i_x is the ***start of replica*** $\mathbf{R}_\rho$ if i_x is the index of the first non-zero row in $\mathbf{R}_\rho$ relative to $\mathbf{R}_r$. We also say that i_y is the ***end of replica*** $\mathbf{R}_\rho$ if i_y is the index of the last non-zero row in $\mathbf{R}_\rho$ relative to $\mathbf{R}_r$. In particular, the start and end of replica $\mathbf{R}_{r+\nu}$ are $\nu\gamma$ and $(m+\nu+1)\gamma - 1$, respectively. For example, the start and end of $\mathbf{R}_r$ are 0 and $(m+1)\gamma - 1$, respectively, regardless from the value of r since $\mathbf{R}_r$ is the reference replica. Moreover, the start and end of $\mathbf{R}_{r+2}$ (resp., $\mathbf{R}_{r-1}$) are 2γ and $(m+3)\gamma - 1$ (resp., $-\gamma$ and $m\gamma - 1$). Furthermore, the indices 1, h, w, and k of replicas are always s.t. $1 < k$ for two replicas, $1 < h < k$ for three replicas, and $1 < h < w < k$ for four replicas.

The counts of different existence possibilities of the nine patterns in addition to the final formulas of F_{P_ℓ}, $\forall \ell$, are presented in the forthcoming subsections. The proofs of all lemmas and theorems in this section are in the appendix of this chapter.

5.4.1 Analysis of Pattern P_1 (size 2×2)

This pattern has two VNs that are *adjacent* (connected via at least one path with only one CN). Thus, Pattern P_1 has its VNs located in at most two replicas, and the pattern spans (i.e., its VNs span) at most $m+1$ consecutive replicas (see [37, Lemma 1]). Suppose P_1 has the CNs c_1 and c_2. The two overlaps forming the pattern are of degree 2, and they are both $c_1 - c_2$ overlaps (among c_1 and c_2).

Lemma 16. *Case 1.1: The number of instances of P_1 with CNs c_1 and c_2, and all overlaps in one replica, $\mathbf{R}_r$, is:*

$$\mathcal{A}_{P_1}\left(t_{\{i_1,i_2\}}\right) = \binom{t_{\{i_1,i_2\}}}{2}. \tag{5.6}$$

Case 1.2: The number of instances of P_1 with CNs c_1 and c_2, and overlaps in two replicas, $\mathbf{R}_r$ and $\mathbf{R}_e$, $r < e$, is:

$$\mathcal{B}_{P_1}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(r-e)\gamma, i_2+(r-e)\gamma\}}\right) = t_{\{i_1,i_2\}} t_{\{i_1+(r-e)\gamma, i_2+(r-e)\gamma\}}. \tag{5.7}$$

The two cases are illustrated in Fig. 5.4.

Theorem 8. *The total number of instances of Pattern P_1 in the binary protograph of an SC code that has parameters $\gamma \geqslant 3$, κ, m, $L \geqslant m+1$, and $\mathcal{O}$, is:*

$$F_{P_1} = \sum_{k=1}^{m+1} (L-k+1) F^k_{P_1,1}, \tag{5.8}$$

where $F^k_{P_1,1}$, $k \in \{1, 2, \ldots, m+1\}$, are given by:

$$F^1_{P_1,1} = \sum_{\{i_1,i_2\} \subset \{0,\ldots,(m+1)\gamma-1\}} \mathcal{A}_{P_1}\left(t_{\{i_1,i_2\}}\right),$$

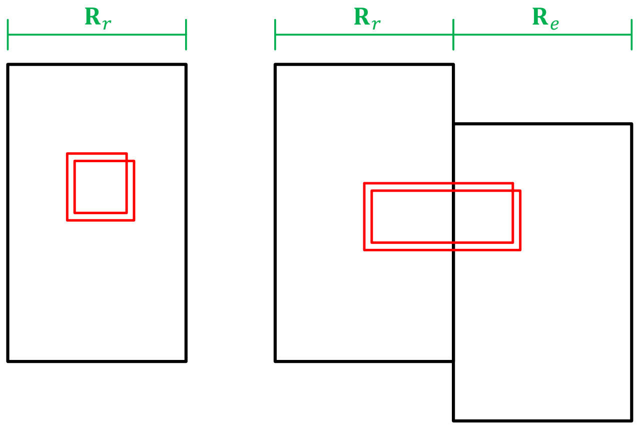


Figure 5.4: An instance of Pattern P_1 in Case 1.1 and in Case 1.2, from left to right. For simplicity, we have $e = r + 1$.

$$F_{P_1,1}^{k\geqslant 2} = \sum_{\{i_1,i_2\}\subset\{(k-1)\gamma,\dots,(m+1)\gamma-1\}} \mathcal{B}_{P_1}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(1-k)\gamma, i_2+(1-k)\gamma\}}\right), \tag{5.9}$$

with $\overline{i_1} \neq \overline{i_2}$*, and* $\overline{i_x}$ *is defined by:* $\overline{i_x} = (i_x \bmod \gamma)$.

5.4.2 Analysis of Pattern P_2 (size 2×3)

This pattern has three VNs, with each pair of them being adjacent. Thus, P_2 spans at most $m + 1$ consecutive replicas. Suppose P_2 has the CNs c_1 and c_2. The three overlaps forming P_2 are of degree 2, and they are all $c_1 - c_2$ overlaps.

Lemma 17. *Case 2.1: The number of instances of P_2 with CNs c_1 and c_2, and all overlaps in one replica, $\mathbf{R}_r$, is:*

$$\mathcal{A}_{P_2}\left(t_{\{i_1,i_2\}}\right) = \binom{t_{\{i_1,i_2\}}}{3}. \tag{5.10}$$

Case 2.2: The number of instances of P_2 with CNs c_1 and c_2, and all overlaps in two replicas s.t. two overlaps are in $\mathbf{R}_r$, and one overlap is in $\mathbf{R}_e$, is:

$$\mathcal{B}_{P_2}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(r-e)\gamma, i_2+(r-e)\gamma\}}\right) = \binom{t_{\{i_1,i_2\}}}{2} t_{\{i_1+(r-e)\gamma, i_2+(r-e)\gamma\}}. \tag{5.11}$$

Case 2.3: The number of instances of P_2 with CNs c_1 and c_2, and overlaps in three replicas

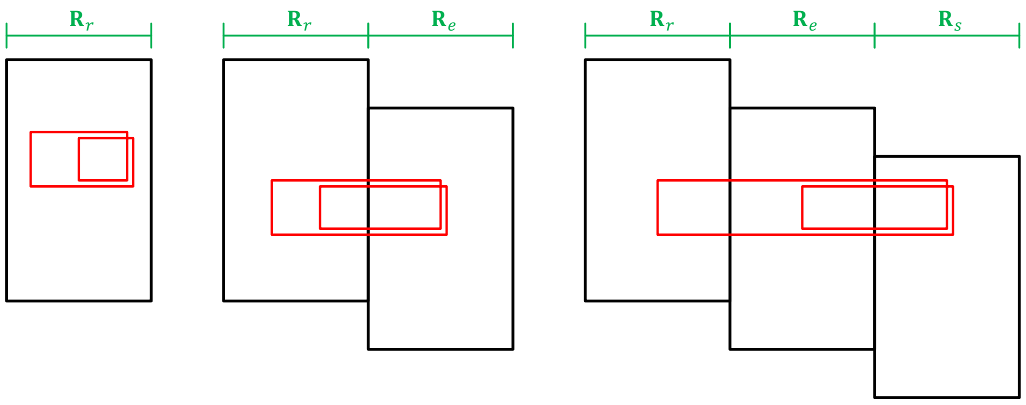


Figure 5.5: An instance of Pattern P_2 in Case 2.1, in Case 2.2, and in Case 2.3, from left to right. For simplicity, we have $e = r + 1$ and $s = e + 1$.

(one in each), $\mathbf{R}_r$, $\mathbf{R}_e$, and $\mathbf{R}_s$, $r < e < s$, is:

$$\begin{aligned}&\mathcal{C}_{P_2}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma\}}, t_{\{i_1+(r-s)\gamma,i_2+(r-s)\gamma\}}\right)\\ &\quad = t_{\{i_1,i_2\}}t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma\}}t_{\{i_1+(r-s)\gamma,i_2+(r-s)\gamma\}}.\end{aligned} \tag{5.12}$$

The three cases are illustrated in Fig 5.5.

Theorem 9. *The total number of instances of Pattern P_2 in the binary protograph of an SC code that has parameters $\gamma \geqslant 3$, κ, m, $L \geqslant m+1$, and $\mathcal{O}$, is:*

$$F_{P_2} = \sum_{k=1}^{m+1} (L-k+1) F^k_{P_2,1}, \tag{5.13}$$

where $F^k_{P_2,1}$, $k \in \{1, 2, \dots, m+1\}$, are given by:

$$\begin{aligned}F^1_{P_2,1} &= \sum_{\{i_1,i_2\}\subset\{0,\dots,(m+1)\gamma-1\}} \mathcal{A}_{P_2}\left(t_{\{i_1,i_2\}}\right),\\ F^2_{P_2,1} &= \sum_{\{i_1,i_2\}\subset\{\gamma,\dots,(m+1)\gamma-1\}} \mathcal{B}_{P_2}\left(t_{\{i_1,i_2\}}, t_{\{i_1-\gamma,i_2-\gamma\}}\right)\\ &\quad + \sum_{\{i_1,i_2\}\subset\{0,\dots,m\gamma-1\}} \mathcal{B}_{P_2}\left(t_{\{i_1,i_2\}}, t_{\{i_1+\gamma,i_2+\gamma\}}\right),\\ F^{k\geqslant 3}_{P_2,1} &= \sum_{\{i_1,i_2\}\subset\{(k-1)\gamma,\dots,(m+1)\gamma-1\}} \mathcal{B}_{P_2}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(1-k)\gamma,i_2+(1-k)\gamma\}}\right)\end{aligned}$$

$$+ \sum_{\{i_1,i_2\}\subset\{0,\dots,(m-k+2)\gamma-1\}} \mathcal{B}_{P_2}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(k-1)\gamma,i_2+(k-1)\gamma\}}\right)$$
$$+ \sum_{h=2}^{k-1} \sum_{\{i_1,i_2\}\subset\{(k-1)\gamma,\dots,(m+1)\gamma-1\}} \mathcal{C}_{P_2}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(1-h)\gamma,i_2+(1-h)\gamma\}}, t_{\{i_1+(1-k)\gamma,i_2+(1-k)\gamma\}}\right), \tag{5.14}$$

with $\overline{i_1} \neq \overline{i_2}$.

5.4.3 Analysis of Pattern P_3 (size 3×2)

This pattern has two VNs that are adjacent. Thus, Pattern P_3 spans at most $m+1$ consecutive replicas. Suppose P_3 has the CNs c_1, c_2, and c_3. The two overlaps forming P_3 are of degree 3, and they are both $c_1 - c_2 - c_3$ overlaps.

Lemma 18. *Case 3.1: The number of instances of* P_3 *with CNs* c_1*,* c_2*, and* c_3*, and all overlaps in one replica,* $\mathbf{R}_r$*, is:*

$$\mathcal{A}_{P_3}\left(t_{\{i_1,i_2,i_3\}}\right) = \binom{t_{\{i_1,i_2,i_3\}}}{2}. \tag{5.15}$$

Case 3.2: The number of instances of P_3 *with CNs* c_1*,* c_2*, and* c_3*, and overlaps in two replicas,* $\mathbf{R}_r$ *and* $\mathbf{R}_e$*,* $r < e$*, is:*

$$\mathcal{B}_{P_3}\left(t_{\{i_1,i_2,i_3\}}, t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma,i_3+(r-e)\gamma\}}\right) = t_{\{i_1,i_2,i_3\}} t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma,i_3+(r-e)\gamma\}}. \tag{5.16}$$

The two cases are illustrated in Fig. 5.6.

Theorem 10. *The total number of instances of Pattern* P_3 *in the binary protograph of an SC code that has parameters* $\gamma \geqslant 3$*,* κ*,* m*,* $L \geqslant m+1$*, and* $\mathcal{O}$*, is:*

$$F_{P_3} = \sum_{k=1}^{m+1} (L-k+1) F_{P_3,1}^{k}, \tag{5.17}$$

where $F_{P_3,1}^{k}$*,* $k \in \{1, 2, \dots, m+1\}$*, are given by:*

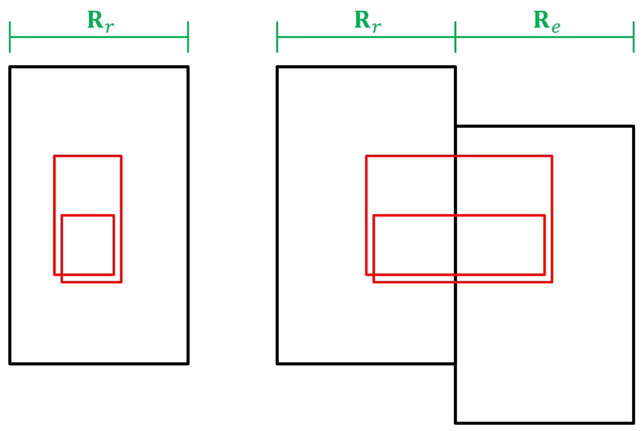


Figure 5.6: An instance of Pattern P_3 in Case 3.1 and in Case 3.2, from left to right. For simplicity, we have $e = r + 1$.

$$F^{1}_{P_3,1} = \sum_{\{i_1,i_2,i_3\}\subset\{0,\dots,(m+1)\gamma-1\}} \mathcal{A}_{P_3}\left(t_{\{i_1,i_2,i_3\}}\right),$$
$$F^{k\geqslant 2}_{P_3,1} = \sum_{\{i_1,i_2,i_3\}\subseteq\{(k-1)\gamma,\dots,(m+1)\gamma-1\}} \mathcal{B}_{P_3}\left(t_{\{i_1,i_2,i_3\}}, t_{\{i_1+(1-k)\gamma,i_2+(1-k)\gamma,i_3+(1-k)\gamma\}}\right), \tag{5.18}$$

with $\overline{i_1} \neq \overline{i_2}$, $\overline{i_1} \neq \overline{i_3}$, *and* $\overline{i_2} \neq \overline{i_3}$.

5.4.4 Analysis of Pattern P_4 (size 2×4)

This pattern has four VNs, with each pair of them being adjacent. Consequently, P_4 spans at most $m + 1$ consecutive replicas. Suppose P_4 has the CNs c_1 and c_2. The four overlaps forming P_4 are of degree 2, and they are all $c_1 - c_2$ overlaps.

Lemma 19. *Case 4.1: The number of instances of P_4 with CNs c_1 and c_2, and all overlaps in one replica, $\mathbf{R}_r$, is:*

$$\mathcal{A}_{P_4}\left(t_{\{i_1,i_2\}}\right) = \binom{t_{\{i_1,i_2\}}}{4}. \tag{5.19}$$

Case 4.2: The number of instances of P_4 with CNs c_1 and c_2, and all overlaps in two replicas s.t. three overlaps are in $\mathbf{R}_r$, and one overlap is in $\mathbf{R}_e$, is:

$$\mathcal{B}_{P_4}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma\}}\right) = \binom{t_{\{i_1,i_2\}}}{3} t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma\}}. \tag{5.20}$$

Case 4.3: The number of instances of P_4 with CNs c_1 and c_2, and all overlaps in two replicas s.t. two overlaps are in $\mathbf{R}_r$, and two overlaps are in $\mathbf{R}_e$, $r < e$, is:

$$\mathcal{C}_{P_4}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma\}}\right) = \binom{t_{\{i_1,i_2\}}}{2}\binom{t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma\}}}{2}. \tag{5.21}$$

Case 4.4: The number of instances of P_4 with CNs c_1 and c_2, and all overlaps in three replicas s.t. two overlaps are in $\mathbf{R}_r$, one overlap is in $\mathbf{R}_e$, and one overlap is in $\mathbf{R}_s$, $e < s$, is:

$$\begin{aligned}&\mathcal{D}_{P_4}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma\}}, t_{\{i_1+(r-s)\gamma,i_2+(r-s)\gamma\}}\right)\\ &\quad= \binom{t_{\{i_1,i_2\}}}{2} t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma\}} t_{\{i_1+(r-s)\gamma,i_2+(r-s)\gamma\}}.\end{aligned} \tag{5.22}$$

Case 4.5: The number of instances of P_4 with CNs c_1 and c_2, and overlaps in four replicas, $\mathbf{R}_r$, $\mathbf{R}_e$, $\mathbf{R}_s$, and $\mathbf{R}_u$, $r < e < s < u$, is:

$$\begin{aligned}&\mathcal{E}_{P_4}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma\}}, t_{\{i_1+(r-s)\gamma,i_2+(r-s)\gamma\}}, t_{\{i_1+(r-u)\gamma,i_2+(r-u)\gamma\}}\right)\\ &\quad= t_{\{i_1,i_2\}} t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma\}} t_{\{i_1+(r-s)\gamma,i_2+(r-s)\gamma\}} t_{\{i_1+(r-u)\gamma,i_2+(r-u)\gamma\}}.\end{aligned} \tag{5.23}$$

Four of the five cases are illustrated in Fig. 5.7.

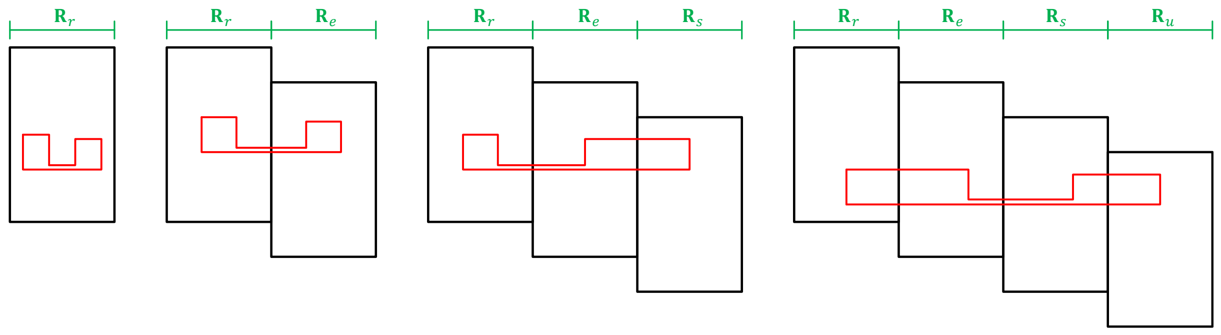


Figure 5.7: An instance of Pattern P_4 in Case 4.1, in Case 4.3, in Case 4.4, and in Case 4.5, from left to right. For simplicity, we have $e = r + 1$, $s = e + 1$, and $u = s + 1$.

Theorem 11. *The total number of instances of Pattern P_4 in the binary protograph of an SC code that has parameters $\gamma \geqslant 3$, κ, m, $L \geqslant m+1$, and $\mathcal{O}$, is:*

$$F_{P_4} = \sum_{k=1}^{m+1} (L-k+1) F^k_{P_4,1}, \tag{5.24}$$

where $F^k_{P_4,1}$, $k \in \{1, 2, \ldots, m+1\}$, are given by:

$$\begin{aligned}
F^1_{P_4,1} &= \sum_{\{i_1,i_2\} \subset \{0,\ldots,(m+1)\gamma-1\}} \mathcal{A}_{P_4}\left(t_{\{i_1,i_2\}}\right),\\
F^2_{P_4,1} &= \sum_{\{i_1,i_2\} \subset \{\gamma,\ldots,(m+1)\gamma-1\}} \mathcal{B}_{P_4}\left(t_{\{i_1,i_2\}}, t_{\{i_1-\gamma,i_2-\gamma\}}\right)\\
&+ \sum_{\{i_1,i_2\} \subset \{0,\ldots,m\gamma-1\}} \mathcal{B}_{P_4}\left(t_{\{i_1,i_2\}}, t_{\{i_1+\gamma,i_2+\gamma\}}\right)\\
&+ \sum_{\{i_1,i_2\} \subset \{\gamma,\ldots,(m+1)\gamma-1\}} \mathcal{C}_{P_4}\left(t_{\{i_1,i_2\}}, t_{\{i_1-\gamma,i_2-\gamma\}}\right),\\
F^3_{P_4,1} &= \sum_{\{i_1,i_2\} \subset \{2\gamma,\ldots,(m+1)\gamma-1\}} \mathcal{B}_{P_4}\left(t_{\{i_1,i_2\}}, t_{\{i_1-2\gamma,i_2-2\gamma\}}\right)\\
&+ \sum_{\{i_1,i_2\} \subset \{0,\ldots,(m-1)\gamma-1\}} \mathcal{B}_{P_4}\left(t_{\{i_1,i_2\}}, t_{\{i_1+2\gamma,i_2+2\gamma\}}\right)\\
&+ \sum_{\{i_1,i_2\} \subset \{2\gamma,\ldots,(m+1)\gamma-1\}} \mathcal{C}_{P_4}\left(t_{\{i_1,i_2\}}, t_{\{i_1-2\gamma,i_2-2\gamma\}}\right)\\
&+ \sum_{\{i_1,i_2\} \subset \{2\gamma,\ldots,(m+1)\gamma-1\}} \mathcal{D}_{P_4}\left(t_{\{i_1,i_2\}}, t_{\{i_1-\gamma,i_2-\gamma\}}, t_{\{i_1-2\gamma,i_2-2\gamma\}}\right)\\
&+ \sum_{\{i_1,i_2\} \subset \{\gamma,\ldots,m\gamma-1\}} \mathcal{D}_{P_4}\left(t_{\{i_1,i_2\}}, t_{\{i_1+\gamma,i_2+\gamma\}}, t_{\{i_1-\gamma,i_2-\gamma\}}\right)\\
&+ \sum_{\{i_1,i_2\} \subset \{0,\ldots,(m-1)\gamma-1\}} \mathcal{D}_{P_4}\left(t_{\{i_1,i_2\}}, t_{\{i_1+2\gamma,i_2+2\gamma\}}, t_{\{i_1+\gamma,i_2+\gamma\}}\right),\\
F^{k\geqslant 4}_{P_4,1} &= \sum_{\{i_1,i_2\} \subset \{(k-1)\gamma,\ldots,(m+1)\gamma-1\}} \mathcal{B}_{P_4}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(1-k)\gamma,i_2+(1-k)\gamma\}}\right)\\
&+ \sum_{\{i_1,i_2\} \subset \{0,\ldots,(m-k+2)\gamma-1\}} \mathcal{B}_{P_4}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(k-1)\gamma,i_2+(k-1)\gamma\}}\right)\\
&+ \sum_{\{i_1,i_2\} \subset \{(k-1)\gamma,\ldots,(m+1)\gamma-1\}} \mathcal{C}_{P_4}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(1-k)\gamma,i_2+(1-k)\gamma\}}\right)
\end{aligned}$$

$$+ \sum_{h=2}^{k-1} \sum_{\{i_1,i_2\}\subset\{(k-1)\gamma,\dots,(m+1)\gamma-1\}} \mathcal{D}_{P_4}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(1-h)\gamma,i_2+(1-h)\gamma\}}, t_{\{i_1+(1-k)\gamma,i_2+(1-k)\gamma\}}\right)$$
$$+ \sum_{h=2}^{k-1} \sum_{\{i_1,i_2\}\subset\{(k-h)\gamma,\dots,(m-h+2)\gamma-1\}} \mathcal{D}_{P_4}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(h-1)\gamma,i_2+(h-1)\gamma\}}, t_{\{i_1+(h-k)\gamma,i_2+(h-k)\gamma\}}\right)$$
$$+ \sum_{h=2}^{k-1} \sum_{\{i_1,i_2\}\subset\{0,\dots,(m-k+2)\gamma-1\}} \mathcal{D}_{P_4}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(k-1)\gamma,i_2+(k-1)\gamma\}}, t_{\{i_1+(k-h)\gamma,i_2+(k-h)\gamma\}}\right)$$
$$+ \sum_{h=2}^{k-2} \sum_{w=h+1}^{k-1} \sum_{\{i_1,i_2\}\subset\{(k-1)\gamma,\dots,(m+1)\gamma-1\}} \mathcal{E}_{P_4}\Big(t_{\{i_1,i_2\}}, t_{\{i_1+(1-h)\gamma,i_2+(1-h)\gamma\}}, t_{\{i_1+(1-w)\gamma,i_2+(1-w)\gamma\}}$$
$$, t_{\{i_1+(1-k)\gamma,i_2+(1-k)\gamma\}}\Big), \tag{5.25}$$

with $\overline{i_1} \neq \overline{i_2}$.

5.4.5 Analysis of Pattern P_5 (size 4×2)

This pattern has two adjacent VNs. Thus, Pattern P_5 spans at most $m+1$ consecutive replicas. Pattern P_5 does not exist in the case of $\gamma = 3$. Suppose P_5 has the CNs c_1, c_2, c_3, and c_4. The two overlaps forming P_5 are of degree 4, and they are both $c_1 - c_2 - c_3 - c_4$ overlaps.

Lemma 20. *Case 5.1: The number of instances of* P_5 *with CNs* c_1, c_2, c_3, *and* c_4, *and all overlaps in one replica,* $\mathbf{R}_r$, *is:*

$$\mathcal{A}_{P_5}\left(t_{\{i_1,i_2,i_3,i_4\}}\right) = \binom{t_{\{i_1,i_2,i_3,i_4\}}}{2}. \tag{5.26}$$

Case 5.2: The number of instances of P_5 *with* c_1, c_2, c_3, *and* c_4, *and overlaps in two replicas,* $\mathbf{R}_r$ *and* $\mathbf{R}_e$, $r < e$, *is:*

$$\mathcal{B}_{P_5}\left(t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma,i_3+(r-e)\gamma,i_4+(r-e)\gamma\}}\right)$$
$$= t_{\{i_1,i_2,i_3,i_4\}} t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma,i_3+(r-e)\gamma,i_4+(r-e)\gamma\}}. \tag{5.27}$$

The two cases are illustrated in Fig. 5.8.

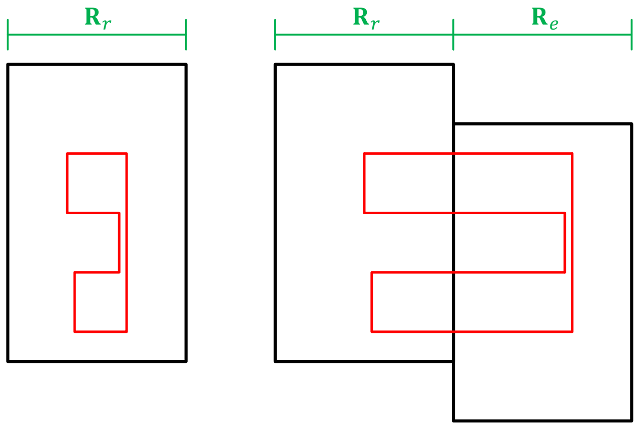


Figure 5.8: An instance of Pattern P_5 in Case 5.1 and in Case 5.2, from left to right. For simplicity, we have $e = r + 1$.

Theorem 12. *The total number of instances of Pattern P_5 in the binary protograph of an SC code that has parameters $\gamma \geqslant 4$, κ, m, $L \geqslant m+1$, and $\mathcal{O}$, is:*

$$F_{P_5} = \sum_{k=1}^{m+1} (L-k+1) F^k_{P_5,1}, \tag{5.28}$$

where $F^k_{P_5,1}$, $k \in \{1, 2, \dots, m+1\}$, are given by:

$$\begin{aligned} F^1_{P_5,1} &= \sum_{\{i_1,i_2,i_3,i_4\} \subset \{0,\dots,(m+1)\gamma-1\}} \mathcal{A}_{P_5}\left(t_{\{i_1,i_2,i_3,i_4\}}\right), \\ F^{k\geqslant 2}_{P_5,1} &= \sum_{\{i_1,i_2,i_3,i_4\} \subseteq \{(k-1)\gamma,\dots,(m+1)\gamma-1\}} \mathcal{B}_{P_5}\left(t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_1+(1-k)\gamma, i_2+(1-k)\gamma, i_3+(1-k)\gamma, i_4+(1-k)\gamma\}}\right), \end{aligned} \tag{5.29}$$

with $\overline{i_1} \neq \overline{i_2}$, $\overline{i_1} \neq \overline{i_3}$, $\overline{i_1} \neq \overline{i_4}$, $\overline{i_2} \neq \overline{i_3}$, $\overline{i_2} \neq \overline{i_4}$, and $\overline{i_3} \neq \overline{i_4}$.

5.4.6 Analysis of Pattern P_6 (size 3×3)

This pattern has three VNs, with each pair of them being adjacent. Thus, P_6 spans at most $m+1$ consecutive replicas. Suppose P_6 has the CNs c_1, c_2, and c_3. Define ***distinct overlaps*** to be overlaps from different families, i.e., overlaps among different sets of CNs. Pattern P_6 is formed of three overlaps; two (distinct) of degree-2 and one of degree-3. Define c_1 as the

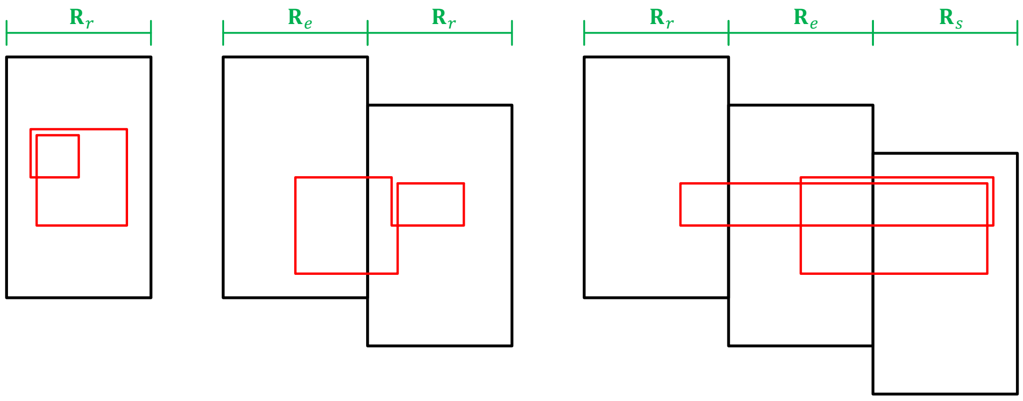


Figure 5.9: An instance of Pattern P_6 in Case 6.1, in Case 6.3, and in Case 6.4, from left to right. For simplicity, we have $e = r + y$, where $y \in \{-1, 1\}$, and $s = e + 1$.

CN connecting the three VNs. Thus, the overlaps are $c_1 - c_2$, $c_1 - c_3$, and $c_1 - c_2 - c_3$ (see P_6 in Fig. 5.3). Again, each VN corresponds to an overlap.

Lemma 21. *Case 6.1: The number of instances of P_6 with CNs c_1, c_2, and c_3 as defined in the previous paragraph, and all overlaps in one replica, $\mathbf{R}_r$, is:*

$$\begin{aligned}\mathcal{A}_{P_6}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_3\}}, t_{\{i_1,i_2,i_3\}}\right) &= t_{\{i_1,i_2,i_3\}}\left(t_{\{i_1,i_2,i_3\}} - 1\right)^+\left(t_{\{i_1,i_3\}} - 2\right)^+ \\ &+ t_{\{i_1,i_2,i_3\}}\left(t_{\{i_1,i_2\}} - t_{\{i_1,i_2,i_3\}}\right)\left(t_{\{i_1,i_3\}} - 1\right)^+. \end{aligned} \tag{5.30}$$

Case 6.2: The number of instances of P_6 with CNs c_1, c_2, and c_3 as defined in the previous paragraph, and all overlaps in two replicas s.t. the two degree-2 overlaps are in $\mathbf{R}_r$, and the degree-3 overlap is in $\mathbf{R}_e$, is:

$$\begin{aligned}&\mathcal{B}_{P_6}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_3\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma,i_3+(r-e)\gamma\}}\right) \\ &\quad = \left[t_{\{i_1,i_2,i_3\}}\left(t_{\{i_1,i_3\}} - 1\right)^+ + \left(t_{\{i_1,i_2\}} - t_{\{i_1,i_2,i_3\}}\right)t_{\{i_1,i_3\}}\right]t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma,i_3+(r-e)\gamma\}}.\end{aligned} \tag{5.31}$$

Case 6.3: The number of instances of P_6 with CNs c_1, c_2, and c_3 as defined in the previous paragraph, and all overlaps in two replicas s.t. the degree-3 overlap and the $c_1 - c_2$ overlap are in $\mathbf{R}_r$, and the $c_1 - c_3$ overlap is in $\mathbf{R}_e$, is:

$$\mathcal{C}_{P_6}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1+(r-e)\gamma, i_3+(r-e)\gamma\}}\right) = t_{\{i_1,i_2,i_3\}}\left(t_{\{i_1,i_2\}}-1\right)^{+} t_{\{i_1+(r-e)\gamma, i_3+(r-e)\gamma\}}. \tag{5.32}$$

Case 6.4: The number of instances of P_6 with CNs c_1, c_2, and c_3 as defined in the previous paragraph, and overlaps in three replicas s.t. the $c_1 - c_2$ overlap is in $\mathbf{R}_r$, the $c_1 - c_3$ overlap is in $\mathbf{R}_e$, and the degree-3 overlap is in $\mathbf{R}_s$, $r < e$, is:

$$\begin{aligned}&\mathcal{D}_{P_6}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(r-e)\gamma, i_3+(r-e)\gamma\}}, t_{\{i_1+(r-s)\gamma, i_2+(r-s)\gamma, i_3+(r-s)\gamma\}}\right)\\ &\quad = t_{\{i_1,i_2\}} t_{\{i_1+(r-e)\gamma, i_3+(r-e)\gamma\}} t_{\{i_1+(r-s)\gamma, i_2+(r-s)\gamma, i_3+(r-s)\gamma\}}.\end{aligned} \tag{5.33}$$

Three of the four cases are illustrated in Fig. 5.9.

Theorem 13. *The total number of instances of Pattern P_6 in the binary protograph of an SC code that has parameters $\gamma \geqslant 3$, κ, m, $L \geqslant m+1$, and $\mathcal{O}$, is:*

$$F_{P_6} = \sum_{k=1}^{m+1} (L-k+1) F^{k}_{P_6,1}, \tag{5.34}$$

where $F^{k}_{P_6,1}$, $k \in \{1,2,\dots,m+1\}$, are given by:

$$\begin{aligned}
F^{1}_{P_6,1} =& \sum_{i_1\in\{0,\dots,(m+1)\gamma-1\}, \{i_2,i_3\}\subset\{0,\dots,(m+1)\gamma-1\}} \mathcal{A}_{P_6}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_3\}}, t_{\{i_1,i_2,i_3\}}\right),\\
F^{2}_{P_6,1} =& \sum_{i_1\in\{\gamma,\dots,(m+1)\gamma-1\}, \{i_2,i_3\}\subset\{\gamma,\dots,(m+1)\gamma-1\}} \mathcal{B}_{P_6}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_3\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1-\gamma,i_2-\gamma,i_3-\gamma\}}\right)\\
&+ \sum_{i_1\in\{0,\dots,m\gamma-1\}, \{i_2,i_3\}\subset\{0,\dots,m\gamma-1\}} \mathcal{B}_{P_6}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_3\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1+\gamma,i_2+\gamma,i_3+\gamma\}}\right)\\
&+ \sum_{i_1\in\{\gamma,\dots,(m+1)\gamma-1\}, i_2\in\{0,\dots,(m+1)\gamma-1\}, i_3\in\{\gamma,\dots,(m+1)\gamma-1\}} \mathcal{C}_{P_6}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1-\gamma,i_3-\gamma\}}\right)\\
&+ \sum_{i_1\in\{0,\dots,m\gamma-1\}, i_2\in\{0,\dots,(m+1)\gamma-1\}, i_3\in\{0,\dots,m\gamma-1\}} \mathcal{C}_{P_6}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1+\gamma,i_3+\gamma\}}\right),\\
F^{k\geqslant 3}_{P_6,1} =& \sum_{i_1\in\{(k-1)\gamma,\dots,(m+1)\gamma-1\}, \{i_2,i_3\}\subset\{(k-1)\gamma,\dots,(m+1)\gamma-1\}} \mathcal{B}_{P_6}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_3\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1+(1-k)\gamma,i_2+(1-k)\gamma,i_3+(1-k)\gamma\}}\right)
\end{aligned}$$

$$+ \sum_{i_1 \in \{0,\ldots,(m-k+2)\gamma-1\}, \{i_2,i_3\} \subset \{0,\ldots,(m-k+2)\gamma-1\}} \mathcal{B}_{P_6}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_3\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1+(k-1)\gamma, i_2+(k-1)\gamma, i_3+(k-1)\gamma\}}\right)$$

$$+ \sum_{i_1 \in \{(k-1)\gamma,\ldots,(m+1)\gamma-1\}, i_2 \in \{0,\ldots,(m+1)\gamma-1\}, i_3 \in \{(k-1)\gamma,\ldots,(m+1)\gamma-1\}} \mathcal{C}_{P_6}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1+(1-k)\gamma, i_3+(1-k)\gamma\}}\right)$$

$$+ \sum_{i_1 \in \{0,\ldots,(m-k+2)\gamma-1\}, i_2 \in \{0,\ldots,(m+1)\gamma-1\}, i_3 \in \{0,\ldots,(m-k+2)\gamma-1\}} \mathcal{C}_{P_6}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1+(k-1)\gamma, i_3+(k-1)\gamma\}}\right)$$

$$+ \sum_{h=2}^{k-1} \sum_{i_1 \in \{(k-1)\gamma,\ldots,(m+1)\gamma-1\}, i_2 \in \{(k-1)\gamma,\ldots,(m+1)\gamma-1\}, i_3 \in \{(k-1)\gamma,\ldots,(m+h)\gamma-1\}} \mathcal{D}_{P_6}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(1-h)\gamma, i_3+(1-h)\gamma\}}, t_{\{i_1+(1-k)\gamma, i_2+(1-k)\gamma, i_3+(1-k)\gamma\}}\right)$$

$$+ \sum_{h=2}^{k-1} \sum_{i_1 \in \{(k-1)\gamma,\ldots,(m+1)\gamma-1\}, i_2 \in \{(h-1),\ldots,(m+1)\gamma-1\}, i_3 \in \{(k-1)\gamma,\ldots,(m+h)\gamma-1\}} \mathcal{D}_{P_6}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(1-k)\gamma, i_3+(1-k)\gamma\}}, t_{\{i_1+(1-h)\gamma, i_2+(1-h)\gamma, i_3+(1-h)\gamma\}}\right)$$

$$+ \sum_{h=2}^{k-1} \sum_{i_1 \in \{(k-h)\gamma,\ldots,(m-h+2)\gamma-1\}, i_2 \in \{0,\ldots,(m-h+2)\gamma-1\}, i_3 \in \{(k-h)\gamma,\ldots,(m-h+2)\gamma-1\}} \mathcal{D}_{P_6}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(h-k)\gamma, i_3+(h-k)\gamma\}}, t_{\{i_1+(h-1)\gamma, i_2+(h-1)\gamma, i_3+(h-1)\gamma\}}\right), \quad (5.35)$$

with $\overline{i_1} \neq \overline{i_2}$*,* $\overline{i_1} \neq \overline{i_3}$*, and* $\overline{i_2} \neq \overline{i_3}$*.*

5.4.7 Analysis of Pattern P_7 (size 3×4)

This pattern has four VNs, with each pair of them being adjacent. Consequently, P_7 spans at most $m+1$ consecutive replicas. Suppose P_7 has the CNs c_1, c_2, and c_3. The pattern is formed of four degree-2 overlaps that are evenly distributed over two different families. Define c_1 as the CN connecting the four VNs. Thus, the overlaps are two $c_1 - c_2$ and two $c_1 - c_3$ overlaps (see P_7 in Fig. 5.3 for clarification).

Lemma 22. *Case 7.1: The number of instances of* P_7 *with CNs* c_1*,* c_2*, and* c_3 *as defined in the previous paragraph, and all overlaps in one replica,* $\mathbf{R}_r$*, is:*

$$\mathcal{A}_{P_7}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_3\}}, t_{\{i_1,i_2,i_3\}}\right) = \binom{t_{\{i_1,i_2,i_3\}}}{2} \binom{\left(t_{\{i_1,i_3\}} - 2\right)^+}{2}$$
$$+ t_{\{i_1,i_2,i_3\}} \left(t_{\{i_1,i_2\}} - t_{\{i_1,i_2,i_3\}}\right) \binom{\left(t_{\{i_1,i_3\}} - 1\right)^+}{2} + \binom{t_{\{i_1,i_2\}} - t_{\{i_1,i_2,i_3\}}}{2} \binom{t_{\{i_1,i_3\}}}{2}. \quad (5.36)$$

Case 7.2: The number of instances of P_7 *with CNs* c_1*,* c_2*, and* c_3 *as defined in the previous paragraph, and all overlaps in two replicas s.t. three overlaps are in* $\mathbf{R}_r$*, and one* $c_1 - c_3$

overlap is in $\mathbf{R}_e$*, is:*

$$\begin{aligned}
&\mathcal{B}_{P_7}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_3\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1+(r-e)\gamma, i_3+(r-e)\gamma\}}\right) \\
&\quad = \left[\binom{t_{\{i_1,i_2,i_3\}}}{2}\left(t_{\{i_1,i_3\}}-2\right)^+ + t_{\{i_1,i_2,i_3\}}\left(t_{\{i_1,i_2\}}-t_{\{i_1,i_2,i_3\}}\right)\left(t_{\{i_1,i_3\}}-1\right)^+ \right.\\
&\qquad \left. + \binom{t_{\{i_1,i_2\}}-t_{\{i_1,i_2,i_3\}}}{2} t_{\{i_1,i_3\}}\right] t_{\{i_1+(r-e)\gamma, i_3+(r-e)\gamma\}}. \qquad (5.37)
\end{aligned}$$

Case 7.3: The number of instances of P_7 *with CNs* c_1*,* c_2*, and* c_3 *as defined in the previous paragraph, and all overlaps in two replicas s.t. the two* $c_1 - c_2$ *overlaps are in* $\mathbf{R}_r$*, and the two* $c_1 - c_3$ *overlaps are in* $\mathbf{R}_e$*,* $r < e$*, is:*

$$\mathcal{C}_{P_7}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(r-e)\gamma, i_3+(r-e)\gamma\}}\right) = \binom{t_{\{i_1,i_2\}}}{2}\binom{t_{\{i_1+(r-e)\gamma, i_3+(r-e)\gamma\}}}{2}. \qquad (5.38)$$

Case 7.4: The number of instances of P_7 *with CNs* c_1*,* c_2*, and* c_3 *as defined in the previous paragraph, and all overlaps in two replicas s.t. two distinct overlaps (from different families) are in* $\mathbf{R}_r$*, and two distinct overlaps are in* $\mathbf{R}_e$*,* $r < e$*, is:*

$$\begin{aligned}
&\mathcal{D}_{P_7}\Big(t_{\{i_1,i_2\}}, t_{\{i_1,i_3\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1+(r-e)\gamma, i_2+(r-e)\gamma\}}, t_{\{i_1+(r-e)\gamma, i_3+(r-e)\gamma\}} \\
&\quad , t_{\{i_1+(r-e)\gamma, i_2+(r-e)\gamma, i_3+(r-e)\gamma\}}\Big) = \left[t_{\{i_1,i_2,i_3\}}\left(t_{\{i_1,i_3\}}-1\right)^+ + \left(t_{\{i_1,i_2\}}-t_{\{i_1,i_2,i_3\}}\right)t_{\{i_1,i_3\}}\right] \\
&\quad \cdot \left[t_{\{i_1+(r-e)\gamma, i_2+(r-e)\gamma, i_3+(r-e)\gamma\}}\left(t_{\{i_1+(r-e)\gamma, i_3+(r-e)\gamma\}}-1\right)^+ \right.\\
&\qquad \left. + \left(t_{\{i_1+(r-e)\gamma, i_2+(r-e)\gamma\}} - t_{\{i_1+(r-e)\gamma, i_2+(r-e)\gamma, i_3+(r-e)\gamma\}}\right) t_{\{i_1+(r-e)\gamma, i_3+(r-e)\gamma\}}\right]. \qquad (5.39)
\end{aligned}$$

Case 7.5: The number of instances of P_7 *with CNs* c_1*,* c_2*, and* c_3 *as defined in the previous paragraph, and all overlaps in three replicas s.t. the two* $c_1 - c_2$ *overlaps are in* $\mathbf{R}_r$*, and the* $c_1 - c_3$ *overlaps are in* $\mathbf{R}_e$ *and* $\mathbf{R}_s$*,* $e < s$*, is:*

$$\begin{aligned}
&\mathcal{E}_{P_7}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(r-e)\gamma, i_3+(r-e)\gamma\}}, t_{\{i_1+(r-s)\gamma, i_3+(r-s)\gamma\}}\right) \\
&\quad = \binom{t_{\{i_1,i_2\}}}{2} t_{\{i_1+(r-e)\gamma, i_3+(r-e)\gamma\}} t_{\{i_1+(r-s)\gamma, i_3+(r-s)\gamma\}}. \qquad (5.40)
\end{aligned}$$

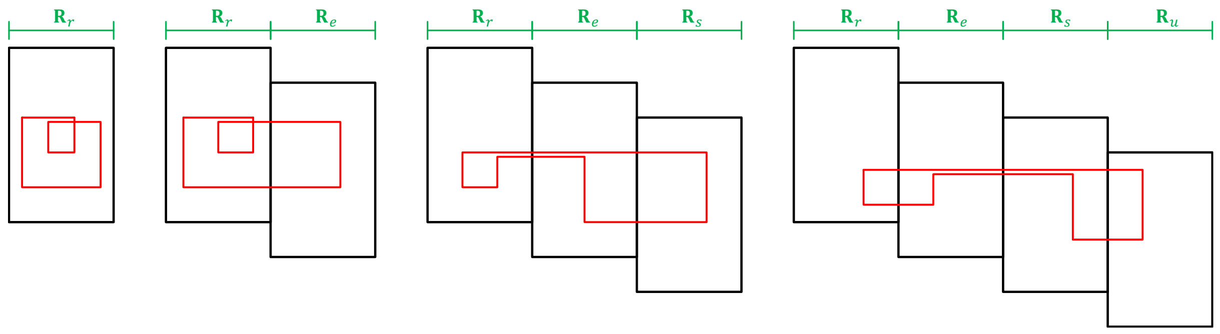


Figure 5.10: An instance of Pattern P_7 in Case 7.1, in Case 7.2, in Case 7.5, and in Case 7.7, from left to right. For simplicity, we have $e = r+1$, $s = e+1$, and $u = s+1$.

Case 7.6: The number of instances of P_7 with CNs c_1, c_2, and c_3 as defined in the previous paragraph, and all overlaps in three replicas s.t. two distinct overlaps (from different families) are in $\mathbf{R}_r$, one $c_1 - c_2$ overlap is in $\mathbf{R}_e$, and one $c_1 - c_3$ overlap is in $\mathbf{R}_s$, $e < s$, is:

$$\begin{aligned}&\mathcal{G}_{P_7}\Big(t_{\{i_1,i_2\}}, t_{\{i_1,i_3\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1+(r-e)\gamma, i_2+(r-e)\gamma\}}, t_{\{i_1+(r-s)\gamma, i_3+(r-s)\gamma\}}\Big)\\&\quad= \Big[t_{\{i_1,i_2,i_3\}}\left(t_{\{i_1,i_3\}}-1\right)^+ + \left(t_{\{i_1,i_2\}} - t_{\{i_1,i_2,i_3\}}\right) t_{\{i_1,i_3\}}\Big]\\&\qquad \cdot t_{\{i_1+(r-e)\gamma, i_2+(r-e)\gamma\}} t_{\{i_1+(r-s)\gamma, i_3+(r-s)\gamma\}}. \end{aligned} \tag{5.41}$$

Case 7.7: The number of instances of P_7 with CNs c_1, c_2, and c_3 as defined in the previous paragraph, and overlaps in four replicas s.t. the two $c_1 - c_2$ overlaps are in $\mathbf{R}_r$ and $\mathbf{R}_e$, and the two $c_1 - c_3$ overlaps are in $\mathbf{R}_s$ and $\mathbf{R}_u$, $r < e$, $r < s$, and $s < u$, is:

$$\begin{aligned}&\mathcal{I}_{P_7}\left(t_{\{i_1,i_2\}}, t_{\{i_1+(r-e)\gamma, i_2+(r-e)\gamma\}}, t_{\{i_1+(r-s)\gamma, i_3+(r-s)\gamma\}}, t_{\{i_1+(r-u)\gamma, i_3+(r-u)\gamma\}}\right)\\&\quad= t_{\{i_1,i_2\}} t_{\{i_1+(r-e)\gamma, i_2+(r-e)\gamma\}} t_{\{i_1+(r-s)\gamma, i_3+(r-s)\gamma\}} t_{\{i_1+(r-u)\gamma, i_3+(r-u)\gamma\}}. \end{aligned} \tag{5.42}$$

Four of the seven cases are illustrated in Fig. 5.10.

Theorem 14. *The total number of instances of Pattern P_7 in the binary protograph of an SC code that has parameters $\gamma \geqslant 3$, κ, m, $L \geqslant m+1$, and $\mathcal{O}$, is:*

$$F_{P_7} = \sum_{k=1}^{m+1} (L-k+1) F^k_{P_7,1}, \tag{5.43}$$

where $F^k_{P_7,1}$*,* $k \in \{1,2,\ldots,m+1\}$*, are given by:*

$$\begin{aligned}
F^1_{P_7,1} &= \sum_{i_1\in\{0,\ldots,(m+1)\gamma-1\},\{i_2,i_3\}\subset\{0,\ldots,(m+1)\gamma-1\}} \mathcal{A}_{P_7}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_3\}}, t_{\{i_1,i_2,i_3\}}\right),\\
F^2_{P_7,1} &= \sum_{i_1\in\{\gamma,\ldots,(m+1)\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_3\in\{\gamma,\ldots,(m+1)\gamma-1\}} \mathcal{B}_{P_7}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_3\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1-\gamma,i_3-\gamma\}}\right)\\
&\quad+ \sum_{i_1\in\{0,\ldots,m\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_3\in\{0,\ldots,m\gamma-1\}} \mathcal{B}_{P_7}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_3\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1+\gamma,i_3+\gamma\}}\right)\\
&\quad+ \sum_{i_1\in\{\gamma,\ldots,(m+1)\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_3\in\{\gamma,\ldots,(m+2)\gamma-1\}} \mathcal{C}_{P_7}\left(t_{\{i_1,i_2\}}, t_{\{i_1-\gamma,i_3-\gamma\}}\right)\\
&\quad+ \sum_{i_1\in\{\gamma,\ldots,(m+1)\gamma-1\},\{i_2,i_3\}\subset\{\gamma,\ldots,(m+1)\gamma-1\}} \mathcal{D}_{P_7}\Big(t_{\{i_1,i_2\}}, t_{\{i_1,i_3\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1-\gamma,i_2-\gamma\}}, t_{\{i_1-\gamma,i_3-\gamma\}}, t_{\{i_1-\gamma,i_2-\gamma,i_3-\gamma\}}\Big),\\
F^3_{P_7,1} &= \sum_{i_1\in\{2\gamma,\ldots,(m+1)\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_3\in\{2\gamma,\ldots,(m+1)\gamma-1\}} \mathcal{B}_{P_7}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_3\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1-2\gamma,i_3-2\gamma\}}\right)\\
&\quad+ \sum_{i_1\in\{0,\ldots,(m-1)\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_3\in\{0,\ldots,(m-1)\gamma-1\}} \mathcal{B}_{P_7}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_3\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1+2\gamma,i_3+2\gamma\}}\right)\\
&\quad+ \sum_{i_1\in\{2\gamma,\ldots,(m+1)\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_3\in\{2\gamma,\ldots,(m+3)\gamma-1\}} \mathcal{C}_{P_7}\left(t_{\{i_1,i_2\}}, t_{\{i_1-2\gamma,i_3-2\gamma\}}\right)\\
&\quad+ \sum_{i_1\in\{2\gamma,\ldots,(m+1)\gamma-1\},\{i_2,i_3\}\subset\{2\gamma,\ldots,(m+1)\gamma-1\}} \mathcal{D}_{P_7}\Big(t_{\{i_1,i_2\}}, t_{\{i_1,i_3\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1-2\gamma,i_2-2\gamma\}}, t_{\{i_1-2\gamma,i_3-2\gamma\}}, t_{\{i_1-2\gamma,i_2-2\gamma,i_3-2\gamma\}}\Big)\\
&\quad+ \sum_{i_1\in\{2\gamma,\ldots,(m+1)\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_3\in\{2\gamma,\ldots,(m+2)\gamma-1\}} \mathcal{E}_{P_7}\Big(t_{\{i_1,i_2\}}, t_{\{i_1-\gamma,i_3-\gamma\}}, t_{\{i_1-2\gamma,i_3-2\gamma\}}\Big)\\
&\quad+ \sum_{i_1\in\{\gamma,\ldots,m\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_3\in\{\gamma,\ldots,m\gamma-1\}} \mathcal{E}_{P_7}\Big(t_{\{i_1,i_2\}}, t_{\{i_1+\gamma,i_3+\gamma\}}, t_{\{i_1-\gamma,i_3-\gamma\}}\Big)\\
&\quad+ \sum_{i_1\in\{0,\ldots,(m-1)\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_3\in\{-\gamma,\ldots,(m-1)\gamma-1\}} \mathcal{E}_{P_7}\Big(t_{\{i_1,i_2\}}, t_{\{i_1+2\gamma,i_3+2\gamma\}}, t_{\{i_1+\gamma,i_3+\gamma\}}\Big)\\
&\quad+ \sum_{i_1\in\{2\gamma,\ldots,(m+1)\gamma-1\},i_2\in\{\gamma,\ldots,(m+1)\gamma-1\},i_3\in\{2\gamma,\ldots,(m+1)\gamma-1\}} \mathcal{G}_{P_7}\Big(t_{\{i_1,i_2\}}, t_{\{i_1,i_3\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1-\gamma,i_2-\gamma\}}, t_{\{i_1-2\gamma,i_3-2\gamma\}}\Big)\\
&\quad+ \sum_{i_1\in\{\gamma,\ldots,m\gamma-1\},i_2\in\{0,\ldots,m\gamma-1\},i_3\in\{\gamma,\ldots,(m+1)\gamma-1\}} \mathcal{G}_{P_7}\Big(t_{\{i_1,i_2\}}, t_{\{i_1,i_3\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1+\gamma,i_2+\gamma\}}, t_{\{i_1-\gamma,i_3-\gamma\}}\Big)\\
&\quad+ \sum_{i_1\in\{0,\ldots,(m-1)\gamma-1\},i_2\in\{0,\ldots,(m-1)\gamma-1\},i_3\in\{0,\ldots,m\gamma-1\}} \mathcal{G}_{P_7}\Big(t_{\{i_1,i_2\}}, t_{\{i_1,i_3\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1+2\gamma,i_2+2\gamma\}}, t_{\{i_1+\gamma,i_3+\gamma\}}\Big),
\end{aligned}$$

$$
\begin{aligned}
F_{P_7,1}^{k\geqslant 4} = & \sum_{i_1\in\{(k-1)\gamma,\dots,(m+1)\gamma-1\},i_2\in\{0,\dots,(m+1)\gamma-1\},i_3\in\{(k-1)\gamma,\dots,(m+1)\gamma-1\}} \mathcal{B}_{P_7}\left(t_{\{i_1,i_2\}},t_{\{i_1,i_3\}},t_{\{i_1,i_2,i_3\}},t_{\{i_1+(1-k)\gamma,i_3+(1-k)\gamma\}}\right) \\
& + \sum_{i_1\in\{0,\dots,(m-k+2)\gamma-1\},i_2\in\{0,\dots,(m+1)\gamma-1\},i_3\in\{0,\dots,(m-k+2)\gamma-1\}} \mathcal{B}_{P_7}\left(t_{\{i_1,i_2\}},t_{\{i_1,i_3\}},t_{\{i_1,i_2,i_3\}},t_{\{i_1+(k-1)\gamma,i_3+(k-1)\gamma\}}\right) \\
& + \sum_{i_1\in\{(k-1)\gamma,\dots,(m+1)\gamma-1\},i_2\in\{0,\dots,(m+1)\gamma-1\},i_3\in\{(k-1)\gamma,\dots,(m+k)\gamma-1\}} \mathcal{C}_{P_7}\left(t_{\{i_1,i_2\}},t_{\{i_1+(1-k)\gamma,i_3+(1-k)\gamma\}}\right) \\
& + \sum_{i_1\in\{(k-1)\gamma,\dots,(m+1)\gamma-1\},\{i_2,i_3\}\subset\{(k-1)\gamma,\dots,(m+1)\gamma-1\}} \mathcal{D}_{P_7}\Big(t_{\{i_1,i_2\}},t_{\{i_1,i_3\}},t_{\{i_1,i_2,i_3\}},t_{\{i_1+(1-k)\gamma,i_2+(1-k)\gamma\}},t_{\{i_1+(1-k)\gamma,i_3+(1-k)\gamma\}} \\
& \qquad ,t_{\{i_1+(1-k)\gamma,i_2+(1-k)\gamma,i_3+(1-k)\gamma\}}\Big) \\
& + \sum_{h=2}^{k-1}\sum_{i_1\in\{(k-1)\gamma,\dots,(m+1)\gamma-1\},i_2\in\{0,\dots,(m+1)\gamma-1\},i_3\in\{(k-1)\gamma,\dots,(m+h)\gamma-1\}} \mathcal{E}_{P_7}\Big(t_{\{i_1,i_2\}},t_{\{i_1+(1-h)\gamma,i_3+(1-h)\gamma\}},t_{\{i_1+(1-k)\gamma,i_3+(1-k)\gamma\}}\Big) \\
& + \sum_{h=2}^{k-1}\sum_{i_1\in\{(k-h)\gamma,\dots,(m-h+2)\gamma-1\},i_2\in\{0,\dots,(m+1)\gamma-1\},i_3\in\{(k-h)\gamma,\dots,(m-h+2)\gamma-1\}} \mathcal{E}_{P_7}\Big(t_{\{i_1,i_2\}},t_{\{i_1+(h-1)\gamma,i_3+(h-1)\gamma\}},t_{\{i_1+(h-k)\gamma,i_3+(h-k)\gamma\}}\Big) \\
& + \sum_{h=2}^{k-1}\sum_{i_1\in\{0,\dots,(m-k+2)\gamma-1\},i_2\in\{0,\dots,(m+1)\gamma-1\},i_3\in\{(h-k)\gamma,\dots,(m-k+2)\gamma-1\}} \mathcal{E}_{P_7}\Big(t_{\{i_1,i_2\}},t_{\{i_1+(k-1)\gamma,i_3+(k-1)\gamma\}},t_{\{i_1+(k-h)\gamma,i_3+(k-h)\gamma\}}\Big) \\
& + \sum_{h=2}^{k-1}\sum_{i_1\in\{(k-1)\gamma,\dots,(m+1)\gamma-1\},i_2\in\{(h-1)\gamma,\dots,(m+1)\gamma-1\},i_3\in\{(k-1)\gamma,\dots,(m+1)\gamma-1\}} \mathcal{G}_{P_7}\Big(t_{\{i_1,i_2\}},t_{\{i_1,i_3\}},t_{\{i_1,i_2,i_3\}},t_{\{i_1+(1-h)\gamma,i_2+(1-h)\gamma\}},t_{\{i_1+(1-k)\gamma,i_3+(1-k)\gamma\}}\Big) \\
& + \sum_{h=2}^{k-1}\sum_{i_1\in\{(k-h)\gamma,\dots,(m-h+2)\gamma-1\},i_2\in\{0,\dots,(m-h+2)\gamma-1\},i_3\in\{(k-h)\gamma,\dots,(m+1)\gamma-1\}} \mathcal{G}_{P_7}\Big(t_{\{i_1,i_2\}},t_{\{i_1,i_3\}},t_{\{i_1,i_2,i_3\}},t_{\{i_1+(h-1)\gamma,i_2+(h-1)\gamma\}},t_{\{i_1+(h-k)\gamma,i_3+(h-k)\gamma\}}\Big) \\
& + \sum_{h=2}^{k-1}\sum_{i_1\in\{0,\dots,(m-k+2)\gamma-1\},i_2\in\{0,\dots,(m-k+2)\gamma-1\},i_3\in\{0,\dots,(m-k+h+1)\gamma-1\}} \mathcal{G}_{P_7}\Big(t_{\{i_1,i_2\}},t_{\{i_1,i_3\}},t_{\{i_1,i_2,i_3\}},t_{\{i_1+(k-1)\gamma,i_2+(k-1)\gamma\}},t_{\{i_1+(k-h)\gamma,i_3+(k-h)\gamma\}}\Big) \\
& + \sum_{h=2}^{k-2}\sum_{w=h+1}^{k-1}\sum_{i_1\in\{(k-1)\gamma,\dots,(m+1)\gamma-1\},i_2\in\{(h-1)\gamma,\dots,(m+1)\gamma-1\},i_3\in\{(k-1)\gamma,\dots,(m+w)\gamma-1\}} \mathcal{I}_{P_7}\Big(t_{\{i_1,i_2\}},t_{\{i_1+(1-h)\gamma,i_2+(1-h)\gamma\}},t_{\{i_1+(1-w)\gamma,i_3+(1-w)\gamma\}} \\
& \qquad ,t_{\{i_1+(1-k)\gamma,i_3+(1-k)\gamma\}}\Big) \\
& + \sum_{h=2}^{k-2}\sum_{w=h+1}^{k-1}\sum_{i_1\in\{(k-1)\gamma,\dots,(m+1)\gamma-1\},i_2\in\{(w-1)\gamma,\dots,(m+1)\gamma-1\},i_3\in\{(k-1)\gamma,\dots,(m+h)\gamma-1\}} \mathcal{I}_{P_7}\Big(t_{\{i_1,i_2\}},t_{\{i_1+(1-w)\gamma,i_2+(1-w)\gamma\}},t_{\{i_1+(1-h)\gamma,i_3+(1-h)\gamma\}} \\
& \qquad ,t_{\{i_1+(1-k)\gamma,i_3+(1-k)\gamma\}}\Big) \\
& + \sum_{h=2}^{k-2}\sum_{w=h+1}^{k-1}\sum_{i_1\in\{(k-1)\gamma,\dots,(m+1)\gamma-1\},i_2\in\{(k-1)\gamma,\dots,(m+1)\gamma-1\},i_3\in\{(w-1)\gamma,\dots,(m+h)\gamma-1\}} \mathcal{I}_{P_7}\Big(t_{\{i_1,i_2\}},t_{\{i_1+(1-k)\gamma,i_2+(1-k)\gamma\}},t_{\{i_1+(1-h)\gamma,i_3+(1-h)\gamma\}} \\
& \qquad ,t_{\{i_1+(1-w)\gamma,i_3+(1-w)\gamma\}}\Big), \qquad (5.44)
\end{aligned}
$$

with $\overline{i_1} \neq \overline{i_2}$*,* $\overline{i_1} \neq \overline{i_3}$*, and* $i_2 \neq i_3$*.*

5.4.8 Analysis of Pattern P_8 (size 4×3)

This pattern has three VNs, and the adjacent pairs are $v_1 - v_2$ and $v_1 - v_3$ (not all pairs) according to P_8 in Fig. 5.3. Thus, P_8 spans at most $2m + 1$ consecutive replicas (see [37, Lemma 1]). Pattern P_8 does not exist in the case of $\gamma = 3$. Suppose P_8 has the CNs c_1, c_2, c_3, and c_4. The pattern is formed of three overlaps, two of degree-2 and one of degree-4. The degree-2 overlaps are not only distinct, but also mutually exclusive (i.e., they do not share any CNs). Define the CNs such that c_1 and c_2 are directly connected twice, which is the same for c_3 and c_4. Thus, the overlaps are $c_1 - c_2$, $c_3 - c_4$, and $c_1 - c_2 - c_3 - c_4$ (see also P_8 in Fig. 5.3).

Lemma 23. *Case 8.1: The number of instances of P_8 with CNs c_1, c_2, c_3, and c_4 as defined in the previous paragraph, and all overlaps in one replica, $\mathbf{R}_r$, is:*

$$\begin{aligned} \mathcal{A}_{P_8}\left(t_{\{i_1,i_2\}}, t_{\{i_3,i_4\}}, t_{\{i_1,i_2,i_3,i_4\}}\right) &= t_{\{i_1,i_2,i_3,i_4\}}\left(t_{\{i_1,i_2,i_3,i_4\}} - 1\right)^+\left(t_{\{i_3,i_4\}} - 2\right)^+ \\ &+ t_{\{i_1,i_2,i_3,i_4\}}\left(t_{\{i_1,i_2\}} - t_{\{i_1,i_2,i_3,i_4\}}\right)\left(t_{\{i_3,i_4\}} - 1\right)^+. \end{aligned} \tag{5.45}$$

Case 8.2: The number of instances of P_8 with CNs c_1, c_2, c_3, and c_4 as defined in the previous paragraph, and all overlaps in two replicas s.t. the two degree-2 overlaps are in $\mathbf{R}_r$, and the degree-4 overlap is in $\mathbf{R}_e$, is:

$$\begin{aligned} &\mathcal{B}_{P_8}\left(t_{\{i_1,i_2\}}, t_{\{i_3,i_4\}}, t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma,i_3+(r-e)\gamma,i_4+(r-e)\gamma\}}\right) \\ &\quad= \left[t_{\{i_1,i_2,i_3,i_4\}}\left(t_{\{i_3,i_4\}} - 1\right)^+ + \left(t_{\{i_1,i_2\}} - t_{\{i_1,i_2,i_3,i_4\}}\right)t_{\{i_3,i_4\}}\right] \\ &\qquad \cdot t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma,i_3+(r-e)\gamma,i_4+(r-e)\gamma\}}. \end{aligned} \tag{5.46}$$

Case 8.3: The number of instances of P_8 with CNs c_1, c_2, c_3, and c_4 as defined in the previous paragraph, and all overlaps in two replicas s.t. the degree-4 overlap and the $c_1 - c_2$ overlap are in $\mathbf{R}_r$, and the $c_3 - c_4$ overlap is in $\mathbf{R}_e$, is:

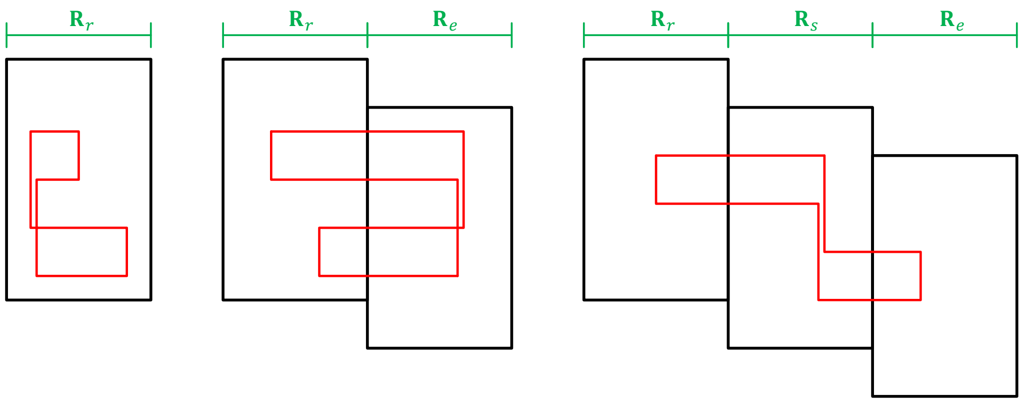


Figure 5.11: An instance of Pattern P_8 in Case 8.1, in Case 8.2, and in Case 8.4, from left to right. For simplicity, we have $e = r + y$, where $y \in \{1, 2\}$, and $s = e - 1$.

$$\mathcal{C}_{P_8}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_3+(r-e)\gamma,i_4+(r-e)\gamma\}}\right) = t_{\{i_1,i_2,i_3,i_4\}}\left(t_{\{i_1,i_2\}} - 1\right)^+ t_{\{i_3+(r-e)\gamma,i_4+(r-e)\gamma\}}. \tag{5.47}$$

Case 8.4: The number of instances of P_8 with c_1, c_2, c_3, and c_4 as defined previously, and overlaps in three replicas s.t. the $c_1 - c_2$ overlap is in $\mathbf{R}_r$, the $c_3 - c_4$ overlap is in $\mathbf{R}_e$, and the degree-4 overlap is in $\mathbf{R}_s$, $r < e$, is:

$$\begin{aligned}&\mathcal{D}_{P_8}\left(t_{\{i_1,i_2\}}, t_{\{i_3+(r-e)\gamma,i_4+(r-e)\gamma\}}, t_{\{i_1+(r-s)\gamma,i_2+(r-s)\gamma,i_3+(r-s)\gamma,i_4+(r-s)\gamma\}}\right)\\&\quad= t_{\{i_1,i_2\}}t_{\{i_3+(r-e)\gamma,i_4+(r-e)\gamma\}}t_{\{i_1+(r-s)\gamma,i_2+(r-s)\gamma,i_3+(r-s)\gamma,i_4+(r-s)\gamma\}}.\end{aligned} \tag{5.48}$$

Three of the four cases are illustrated in Fig. 5.11.

Theorem 15. *The total number of instances of Pattern P_8 in the binary protograph of an SC code that has parameters $\gamma \geqslant 4$, κ, m, $L \geqslant 2m+1$, and $\mathcal{O}$, is:*

$$F_{P_8} = \sum_{k=1}^{2m+1}(L-k+1)F^k_{P_8,1}, \tag{5.49}$$

where $F^k_{P_8,1}$, $k \in \{1, 2, \dots, 2m+1\}$, are given by:

$$F^1_{P_8,1} = \frac{1}{2}\sum_{\{i_1,i_2\}\subset\{0,\dots,(m+1)\gamma-1\},\{i_3,i_4\}\subset\{0,\dots,(m+1)\gamma-1\}} \mathcal{A}_{P_8}\left(t_{\{i_1,i_2\}}, t_{\{i_3,i_4\}}, t_{\{i_1,i_2,i_3,i_4\}}\right),$$

$$
\begin{aligned}
F^{2}_{P_8,1} = {} & \frac{1}{2} \sum_{\{i_1,i_2\}\subset\{\gamma,\dots,(m+1)\gamma-1\},\{i_3,i_4\}\subset\{\gamma,\dots,(m+1)\gamma-1\}} \mathcal{B}_{P_8}\Big(t_{\{i_1,i_2\}}, t_{\{i_3,i_4\}}, t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_1-\gamma,i_2-\gamma,i_3-\gamma,i_4-\gamma\}}\Big) \\
+ {} & \frac{1}{2} \sum_{\{i_1,i_2\}\subset\{0,\dots,m\gamma-1\},\{i_3,i_4\}\subset\{0,\dots,m\gamma-1\}} \mathcal{B}_{P_8}\Big(t_{\{i_1,i_2\}}, t_{\{i_3,i_4\}}, t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_1+\gamma,i_2+\gamma,i_3+\gamma,i_4+\gamma\}}\Big) \\
+ {} & \sum_{\{i_1,i_2\}\subset\{0,\dots,(m+1)\gamma-1\},\{i_3,i_4\}\subset\{\gamma,\dots,(m+1)\gamma-1\}} \mathcal{C}_{P_8}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_3-\gamma,i_4-\gamma\}}\right) \\
+ {} & \sum_{\{i_1,i_2\}\subset\{0,\dots,(m+1)\gamma-1\},\{i_3,i_4\}\subset\{0,\dots,m\gamma-1\}} \mathcal{C}_{P_8}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_3+\gamma,i_4+\gamma\}}\right), \\
F^{k\geqslant 3}_{P_8,1} = {} & \frac{1}{2} \sum_{\{i_1,i_2\}\subset\{(k-1)\gamma,\dots,(m+1)\gamma-1\},\{i_3,i_4\}\subset\{(k-1)\gamma,\dots,(m+1)\gamma-1\}} \mathcal{B}_{P_8}\Big(t_{\{i_1,i_2\}}, t_{\{i_3,i_4\}}, t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_1+(1-k)\gamma,i_2+(1-k)\gamma,i_3+(1-k)\gamma,i_4+(1-k)\gamma\}}\Big) \\
+ {} & \frac{1}{2} \sum_{\{i_1,i_2\}\subset\{0,\dots,(m-k+2)\gamma-1\},\{i_3,i_4\}\subset\{0,\dots,(m-k+2)\gamma-1\}} \mathcal{B}_{P_8}\Big(t_{\{i_1,i_2\}}, t_{\{i_3,i_4\}}, t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_1+(k-1)\gamma,i_2+(k-1)\gamma,i_3+(k-1)\gamma,i_4+(k-1)\gamma\}}\Big) \\
+ {} & \sum_{\{i_1,i_2\}\subset\{0,\dots,(m+1)\gamma-1\},\{i_3,i_4\}\subset\{(k-1)\gamma,\dots,(m+1)\gamma-1\}} \mathcal{C}_{P_8}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_3+(1-k)\gamma,i_4+(1-k)\gamma\}}\right) \\
+ {} & \sum_{\{i_1,i_2\}\subset\{0,\dots,(m+1)\gamma-1\},\{i_3,i_4\}\subset\{0,\dots,(m-k+2)\gamma-1\}} \mathcal{C}_{P_8}\left(t_{\{i_1,i_2\}}, t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_3+(k-1)\gamma,i_4+(k-1)\gamma\}}\right) \\
+ {} & \sum_{h=2}^{k-1} \sum_{\{i_1,i_2\}\subset\{(k-1)\gamma,\dots,(m+1)\gamma-1\},\{i_3,i_4\}\subset\{(k-1)\gamma,\dots,(m+h)\gamma-1\}} \mathcal{D}_{P_8}\Big(t_{\{i_1,i_2\}}, t_{\{i_3+(1-h)\gamma,i_4+(1-h)\gamma\}}, t_{\{i_1+(1-k)\gamma,i_2+(1-k)\gamma,i_3+(1-k)\gamma,i_4+(1-k)\gamma\}}\Big) \\
+ {} & \sum_{h=2}^{k-1} \sum_{\{i_1,i_2\}\subset\{(h-1)\gamma,\dots,(m+1)\gamma-1\},\{i_3,i_4\}\subset\{(k-1)\gamma,\dots,(m+h)\gamma-1\}} \mathcal{D}_{P_8}\Big(t_{\{i_1,i_2\}}, t_{\{i_3+(1-k)\gamma,i_4+(1-k)\gamma\}}, t_{\{i_1+(1-h)\gamma,i_2+(1-h)\gamma,i_3+(1-h)\gamma,i_4+(1-h)\gamma\}}\Big) \\
+ {} & \sum_{h=2}^{k-1} \sum_{\{i_1,i_2\}\subset\{0,\dots,(m-h+2)\gamma-1\},\{i_3,i_4\}\subset\{(k-h)\gamma,\dots,(m-h+2)\gamma-1\}} \mathcal{D}_{P_8}\Big(t_{\{i_1,i_2\}}, t_{\{i_3+(h-k)\gamma,i_4+(h-k)\gamma\}}, t_{\{i_1+(h-1)\gamma,i_2+(h-1)\gamma,i_3+(h-1)\gamma,i_4+(h-1)\gamma\}}\Big),
\end{aligned}
\tag{5.50}
$$

with $\overline{i_1} \neq \overline{i_2}$, $\overline{i_1} \neq \overline{i_3}$, $\overline{i_1} \neq \overline{i_4}$, $\overline{i_2} \neq \overline{i_3}$, $\overline{i_2} \neq \overline{i_4}$, *and* $\overline{i_3} \neq \overline{i_4}$.

5.4.9 Analysis of Pattern P_9 (size 4×4)

This pattern has four VNs, and the adjacent pairs are $v_1 - v_2$, $v_2 - v_3$, $v_3 - v_4$, and $v_1 - v_4$ (not all pairs) according to P_9 in Fig. 5.3. Thus, P_9 also spans at most $2m + 1$ consecutive replicas. Suppose P_9 has the CNs c_1, c_2, c_3, and c_4. The pattern is formed of four distinct degree-2 overlaps. Define the CNs such that the adjacent pairs (connected via at least one path with only one VN) are $c_1 - c_2$, $c_2 - c_3$, $c_3 - c_4$, and $c_1 - c_4$. This definition already implies what the overlaps are.

Lemma 24. *Case 9.1: The number of instances of P_9 with CNs c_1, c_2, c_3, and c_4 as defined in the previous paragraph, and all overlaps in one replica, $\mathbf{R}_r$, is:*

$$\begin{aligned}
&\mathcal{A}_{P_9}\Big(t_{\{i_1,i_2\}}, t_{\{i_2,i_3\}}, t_{\{i_3,i_4\}}, t_{\{i_1,i_4\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1,i_2,i_4\}}, t_{\{i_1,i_3,i_4\}}, t_{\{i_2,i_3,i_4\}}, t_{\{i_1,i_2,i_3,i_4\}}\Big) \\
&\qquad = \mathcal{A}_{P_9,1} + \mathcal{A}_{P_9,2} + \mathcal{A}_{P_9,3} + \mathcal{A}_{P_9,4},
\end{aligned} \tag{5.51}$$

$$\begin{aligned}
\mathcal{A}_{P_9,1} = {} & t_{\{i_1,i_2,i_3,i_4\}} \Big(t_{\{i_1,i_2,i_3,i_4\}} - 1\Big)^+ \Big(t_{\{i_1,i_3,i_4\}} - 2\Big)^+ \Big(t_{\{i_1,i_4\}} - 3\Big)^+ \\
& + t_{\{i_1,i_2,i_3,i_4\}} \Big(t_{\{i_1,i_2,i_3,i_4\}} - 1\Big)^+ \Big(t_{\{i_3,i_4\}} - t_{\{i_1,i_3,i_4\}}\Big) \Big(t_{\{i_1,i_4\}} - 2\Big)^+ \\
& + t_{\{i_1,i_2,i_3,i_4\}} \Big(t_{\{i_2,i_3,i_4\}} - t_{\{i_1,i_2,i_3,i_4\}}\Big) \Big(t_{\{i_1,i_3,i_4\}} - 1\Big)^+ \Big(t_{\{i_1,i_4\}} - 2\Big)^+ \\
& + t_{\{i_1,i_2,i_3,i_4\}} \Big(t_{\{i_2,i_3,i_4\}} - t_{\{i_1,i_2,i_3,i_4\}}\Big) \Big(t_{\{i_3,i_4\}} - t_{\{i_1,i_3,i_4\}} - 1\Big)^+ \Big(t_{\{i_1,i_4\}} - 1\Big)^+ \\
& + t_{\{i_1,i_2,i_3,i_4\}} \Big(t_{\{i_2,i_3\}} - t_{\{i_2,i_3,i_4\}}\Big) \Big(t_{\{i_1,i_3,i_4\}} - 1\Big)^+ \Big(t_{\{i_1,i_4\}} - 2\Big)^+ \\
& + t_{\{i_1,i_2,i_3,i_4\}} \Big(t_{\{i_2,i_3\}} - t_{\{i_2,i_3,i_4\}}\Big) \Big(t_{\{i_3,i_4\}} - t_{\{i_1,i_3,i_4\}}\Big) \Big(t_{\{i_1,i_4\}} - 1\Big)^+,
\end{aligned}$$

$$\begin{aligned}
\mathcal{A}_{P_9,2} = {} & \Big(t_{\{i_1,i_2,i_3\}} - t_{\{i_1,i_2,i_3,i_4\}}\Big) t_{\{i_1,i_2,i_3,i_4\}} \Big(t_{\{i_1,i_3,i_4\}} - 1\Big)^+ \Big(t_{\{i_1,i_4\}} - 2\Big)^+ \\
& + \Big(t_{\{i_1,i_2,i_3\}} - t_{\{i_1,i_2,i_3,i_4\}}\Big) t_{\{i_1,i_2,i_3,i_4\}} \Big(t_{\{i_3,i_4\}} - t_{\{i_1,i_3,i_4\}}\Big) \Big(t_{\{i_1,i_4\}} - 1\Big)^+ \\
& + \Big(t_{\{i_1,i_2,i_3\}} - t_{\{i_1,i_2,i_3,i_4\}}\Big) \Big(t_{\{i_2,i_3,i_4\}} - t_{\{i_1,i_2,i_3,i_4\}}\Big) t_{\{i_1,i_3,i_4\}} \Big(t_{\{i_1,i_4\}} - 1\Big)^+ \\
& + \Big(t_{\{i_1,i_2,i_3\}} - t_{\{i_1,i_2,i_3,i_4\}}\Big) \Big(t_{\{i_2,i_3,i_4\}} - t_{\{i_1,i_2,i_3,i_4\}}\Big) \Big(t_{\{i_3,i_4\}} - t_{\{i_1,i_3,i_4\}} - 1\Big)^+ t_{\{i_1,i_4\}} \\
& + \Big(t_{\{i_1,i_2,i_3\}} - t_{\{i_1,i_2,i_3,i_4\}}\Big) \Big(t_{\{i_2,i_3\}} - t_{\{i_2,i_3,i_4\}} - 1\Big)^+ t_{\{i_1,i_3,i_4\}} \Big(t_{\{i_1,i_4\}} - 1\Big)^+ \\
& + \Big(t_{\{i_1,i_2,i_3\}} - t_{\{i_1,i_2,i_3,i_4\}}\Big) \Big(t_{\{i_2,i_3\}} - t_{\{i_2,i_3,i_4\}} - 1\Big)^+ \Big(t_{\{i_3,i_4\}} - t_{\{i_1,i_3,i_4\}}\Big) t_{\{i_1,i_4\}},
\end{aligned}$$

$$\begin{aligned}
\mathcal{A}_{P_9,3} = {} & \Big(t_{\{i_1,i_2,i_4\}} - t_{\{i_1,i_2,i_3,i_4\}}\Big) t_{\{i_1,i_2,i_3,i_4\}} \Big(t_{\{i_1,i_3,i_4\}} - 1\Big)^+ \Big(t_{\{i_1,i_4\}} - 3\Big)^+ \\
& + \Big(t_{\{i_1,i_2,i_4\}} - t_{\{i_1,i_2,i_3,i_4\}}\Big) t_{\{i_1,i_2,i_3,i_4\}} \Big(t_{\{i_3,i_4\}} - t_{\{i_1,i_3,i_4\}}\Big) \Big(t_{\{i_1,i_4\}} - 2\Big)^+
\end{aligned}$$

$$
\begin{aligned}
&+ \left(t_{\{i_1,i_2,i_4\}} - t_{\{i_1,i_2,i_3,i_4\}}\right)\left(t_{\{i_2,i_3,i_4\}} - t_{\{i_1,i_2,i_3,i_4\}}\right) t_{\{i_1,i_3,i_4\}} \left(t_{\{i_1,i_4\}} - 2\right)^+ \\
&+ \left(t_{\{i_1,i_2,i_4\}} - t_{\{i_1,i_2,i_3,i_4\}}\right)\left(t_{\{i_2,i_3,i_4\}} - t_{\{i_1,i_2,i_3,i_4\}}\right) \\
&\ \cdot \left(t_{\{i_3,i_4\}} - t_{\{i_1,i_3,i_4\}} - 1\right)^+ \left(t_{\{i_1,i_4\}} - 1\right)^+ \\
&+ \left(t_{\{i_1,i_2,i_4\}} - t_{\{i_1,i_2,i_3,i_4\}}\right)\left(t_{\{i_2,i_3\}} - t_{\{i_2,i_3,i_4\}}\right) t_{\{i_1,i_3,i_4\}} \left(t_{\{i_1,i_4\}} - 2\right)^+ \\
&+ \left(t_{\{i_1,i_2,i_4\}} - t_{\{i_1,i_2,i_3,i_4\}}\right)\left(t_{\{i_2,i_3\}} - t_{\{i_2,i_3,i_4\}}\right)\left(t_{\{i_3,i_4\}} - t_{\{i_1,i_3,i_4\}}\right)\left(t_{\{i_1,i_4\}} - 1\right)^+,
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{A}_{P_9,4} = &\left(t_{\{i_1,i_2\}} - t_{\{i_1,i_2,i_3\}} - t_{\{i_1,i_2,i_4\}} + t_{\{i_1,i_2,i_3,i_4\}}\right) t_{\{i_1,i_2,i_3,i_4\}} \\
&\ \cdot \left(t_{\{i_1,i_3,i_4\}} - 1\right)^+ \left(t_{\{i_1,i_4\}} - 2\right)^+ \\
&+ \left(t_{\{i_1,i_2\}} - t_{\{i_1,i_2,i_3\}} - t_{\{i_1,i_2,i_4\}} + t_{\{i_1,i_2,i_3,i_4\}}\right) t_{\{i_1,i_2,i_3,i_4\}} \\
&\ \cdot \left(t_{\{i_3,i_4\}} - t_{\{i_1,i_3,i_4\}}\right)\left(t_{\{i_1,i_4\}} - 1\right)^+ \\
&+ \left(t_{\{i_1,i_2\}} - t_{\{i_1,i_2,i_3\}} - t_{\{i_1,i_2,i_4\}} + t_{\{i_1,i_2,i_3,i_4\}}\right)\left(t_{\{i_2,i_3,i_4\}} - t_{\{i_1,i_2,i_3,i_4\}}\right) \\
&\ \cdot t_{\{i_1,i_3,i_4\}} \left(t_{\{i_1,i_4\}} - 1\right)^+ \\
&+ \left(t_{\{i_1,i_2\}} - t_{\{i_1,i_2,i_3\}} - t_{\{i_1,i_2,i_4\}} + t_{\{i_1,i_2,i_3,i_4\}}\right)\left(t_{\{i_2,i_3,i_4\}} - t_{\{i_1,i_2,i_3,i_4\}}\right) \\
&\ \cdot \left(t_{\{i_3,i_4\}} - t_{\{i_1,i_3,i_4\}} - 1\right)^+ t_{\{i_1,i_4\}} \\
&+ \left(t_{\{i_1,i_2\}} - t_{\{i_1,i_2,i_3\}} - t_{\{i_1,i_2,i_4\}} + t_{\{i_1,i_2,i_3,i_4\}}\right)\left(t_{\{i_2,i_3\}} - t_{\{i_2,i_3,i_4\}}\right) \\
&\ \cdot t_{\{i_1,i_3,i_4\}} \left(t_{\{i_1,i_4\}} - 1\right)^+ \\
&+ \left(t_{\{i_1,i_2\}} - t_{\{i_1,i_2,i_3\}} - t_{\{i_1,i_2,i_4\}} + t_{\{i_1,i_2,i_3,i_4\}}\right)\left(t_{\{i_2,i_3\}} - t_{\{i_2,i_3,i_4\}}\right) \\
&\ \cdot \left(t_{\{i_3,i_4\}} - t_{\{i_1,i_3,i_4\}}\right) t_{\{i_1,i_4\}}. \qquad (5.52)
\end{aligned}
$$

Case 9.2: The number of instances of P_9 with CNs c_1, c_2, c_3, and c_4 as defined in the previous paragraph, and all overlaps in two replicas s.t. three overlaps are in $\mathbf{R}_r$, and the $c_1 - c_4$ overlap is in $\mathbf{R}_e$, is:

$$
\mathcal{B}_{P_9}\Big(t_{\{i_1,i_2\}}, t_{\{i_2,i_3\}}, t_{\{i_3,i_4\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_2,i_3,i_4\}}, t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_1+(r-e)\gamma, i_4+(r-e)\gamma\}}\Big)
$$

$$\begin{aligned}
&= \Big[t_{\{i_1,i_2,i_3,i_4\}} \left(t_{\{i_2,i_3,i_4\}} - 1\right)^+ \left(t_{\{i_3,i_4\}} - 2\right)^+ \\
&\quad + t_{\{i_1,i_2,i_3,i_4\}} \left(t_{\{i_2,i_3\}} - t_{\{i_2,i_3,i_4\}}\right) \left(t_{\{i_3,i_4\}} - 1\right)^+ \\
&\quad + \left(t_{\{i_1,i_2,i_3\}} - t_{\{i_1,i_2,i_3,i_4\}}\right) t_{\{i_2,i_3,i_4\}} \left(t_{\{i_3,i_4\}} - 1\right)^+ \\
&\quad + \left(t_{\{i_1,i_2,i_3\}} - t_{\{i_1,i_2,i_3,i_4\}}\right) \left(t_{\{i_2,i_3\}} - t_{\{i_2,i_3,i_4\}} - 1\right)^+ t_{\{i_3,i_4\}} \\
&\quad + \left(t_{\{i_1,i_2\}} - t_{\{i_1,i_2,i_3\}}\right) t_{\{i_2,i_3,i_4\}} \left(t_{\{i_3,i_4\}} - 1\right)^+ \\
&\quad + \left(t_{\{i_1,i_2\}} - t_{\{i_1,i_2,i_3\}}\right) \left(t_{\{i_2,i_3\}} - t_{\{i_2,i_3,i_4\}}\right) t_{\{i_3,i_4\}}\Big] t_{\{i_1+(r-e)\gamma,i_4+(r-e)\gamma\}}. \qquad (5.53)
\end{aligned}$$

Case 9.3: The number of instances of P_9 with CNs c_1, c_2, c_3, and c_4 as defined in the previous paragraph, and all overlaps in two replicas s.t. $c_1 - c_2$ and $c_2 - c_3$ overlaps are in $\mathbf{R}_r$, and $c_3 - c_4$ and $c_1 - c_4$ overlaps are in $\mathbf{R}_e$, $r < e$, is:

$$\begin{aligned}
&\mathcal{C}_{P_9}\Big(t_{\{i_1,i_2\}}, t_{\{i_2,i_3\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_3+(r-e)\gamma,i_4+(r-e)\gamma\}}, t_{\{i_1+(r-e)\gamma,i_4+(r-e)\gamma\}} \\
&\quad , t_{\{i_1+(r-e)\gamma,i_3+(r-e)\gamma,i_4+(r-e)\gamma\}}\Big) \\
&\quad = \Big[t_{\{i_1,i_2,i_3\}} \left(t_{\{i_2,i_3\}} - 1\right)^+ + \left(t_{\{i_1,i_2\}} - t_{\{i_1,i_2,i_3\}}\right) t_{\{i_2,i_3\}}\Big] \\
&\quad\quad \cdot \Big[t_{\{i_1+(r-e)\gamma,i_3+(r-e)\gamma,i_4+(r-e)\gamma\}} \left(t_{\{i_1+(r-e)\gamma,i_4+(r-e)\gamma\}} - 1\right)^+ \\
&\quad\quad\quad + \left(t_{\{i_3+(r-e)\gamma,i_4+(r-e)\gamma\}} - t_{\{i_1+(r-e)\gamma,i_3+(r-e)\gamma,i_4+(r-e)\gamma\}}\right) t_{\{i_1+(r-e)\gamma,i_4+(r-e)\gamma\}}\Big]. \qquad (5.54)
\end{aligned}$$

Case 9.4: The number of instances of P_9 with CNs c_1, c_2, c_3, and c_4 as defined in the previous paragraph, and all overlaps in two replicas s.t. $c_1 - c_2$ and $c_3 - c_4$ overlaps are in $\mathbf{R}_r$, and $c_2 - c_3$ and $c_1 - c_4$ overlaps are in $\mathbf{R}_e$, $r < e$, is:

$$\begin{aligned}
&\mathcal{D}_{P_9}\Big(t_{\{i_1,i_2\}}, t_{\{i_3,i_4\}}, t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_2+(r-e)\gamma,i_3+(r-e)\gamma\}}, t_{\{i_1+(r-e)\gamma,i_4+(r-e)\gamma\}} \\
&\quad , t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma,i_3+(r-e)\gamma,i_4+(r-e)\gamma\}}\Big) \\
&\quad = \Big[t_{\{i_1,i_2,i_3,i_4\}} \left(t_{\{i_3,i_4\}} - 1\right)^+ + \left(t_{\{i_1,i_2\}} - t_{\{i_1,i_2,i_3,i_4\}}\right) t_{\{i_3,i_4\}}\Big] \\
&\quad\quad \cdot \Big[t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma,i_3+(r-e)\gamma,i_4+(r-e)\gamma\}} \left(t_{\{i_1+(r-e)\gamma,i_4+(r-e)\gamma\}} - 1\right)^+
\end{aligned}$$

$$+ \left(t_{\{i_2+(r-e)\gamma,i_3+(r-e)\gamma\}} - t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma,i_3+(r-e)\gamma,i_4+(r-e)\gamma\}}\right) t_{\{i_1+(r-e)\gamma,i_4+(r-e)\gamma\}}\Bigg]. \tag{5.55}$$

Case 9.5: The number of instances of P_9 with CNs c_1, c_2, c_3, and c_4 as defined previously, and all overlaps in three replicas s.t. $c_1 - c_2$ and $c_2 - c_3$ overlaps are in $\mathbf{R}_r$, the $c_3 - c_4$ overlap is in $\mathbf{R}_e$, and the $c_1 - c_4$ overlap is in $\mathbf{R}_s$, $e < s$, is:

$$\begin{aligned}&\mathcal{E}_{P_9}\left(t_{\{i_1,i_2\}}, t_{\{i_2,i_3\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_3+(r-e)\gamma,i_4+(r-e)\gamma\}}, t_{\{i_1+(r-s)\gamma,i_4+(r-s)\gamma\}}\right)\\ &\quad = \left[t_{\{i_1,i_2,i_3\}}\left(t_{\{i_2,i_3\}} - 1\right)^+ + \left(t_{\{i_1,i_2\}} - t_{\{i_1,i_2,i_3\}}\right) t_{\{i_2,i_3\}}\right]\\ &\qquad \cdot t_{\{i_3+(r-e)\gamma,i_4+(r-e)\gamma\}} t_{\{i_1+(r-s)\gamma,i_4+(r-s)\gamma\}}.\end{aligned} \tag{5.56}$$

Case 9.6: The number of instances of P_9 with CNs c_1, c_2, c_3, and c_4 as defined previously, and all overlaps in three replicas s.t. $c_1 - c_2$ and $c_3 - c_4$ overlaps are in $\mathbf{R}_r$, the $c_2 - c_3$ overlap is in $\mathbf{R}_e$, and the $c_1 - c_4$ overlap is in $\mathbf{R}_s$, $e < s$, is:

$$\begin{aligned}&\mathcal{G}_{P_9}\left(t_{\{i_1,i_2\}}, t_{\{i_3,i_4\}}, t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_2+(r-e)\gamma,i_3+(r-e)\gamma\}}, t_{\{i_1+(r-s)\gamma,i_4+(r-s)\gamma\}}\right)\\ &\quad = \left[t_{\{i_1,i_2,i_3,i_4\}}\left(t_{\{i_3,i_4\}} - 1\right)^+ + \left(t_{\{i_1,i_2\}} - t_{\{i_1,i_2,i_3,i_4\}}\right) t_{\{i_3,i_4\}}\right]\\ &\qquad \cdot t_{\{i_2+(r-e)\gamma,i_3+(r-e)\gamma\}} t_{\{i_1+(r-s)\gamma,i_4+(r-s)\gamma\}}.\end{aligned} \tag{5.57}$$

Case 9.7: The number of instances of P_9 with CNs c_1, c_2, c_3, and c_4 as defined previously, and overlaps in four replicas s.t. the $c_1 - c_2$ overlap is in $\mathbf{R}_r$, the $c_2 - c_3$ overlap is in $\mathbf{R}_e$, the $c_3 - c_4$ overlap is in $\mathbf{R}_s$, and the $c_1 - c_4$ overlap is in $\mathbf{R}_u$, $r < e$, $e < u$, and $r < s$, is:

$$\begin{aligned}&\mathcal{I}_{P_9}\left(t_{\{i_1,i_2\}}, t_{\{i_2+(r-e)\gamma,i_3+(r-e)\gamma\}}, t_{\{i_3+(r-s)\gamma,i_4+(r-s)\gamma\}}, t_{\{i_1+(r-u)\gamma,i_4+(r-u)\gamma\}}\right)\\ &\quad = t_{\{i_1,i_2\}} t_{\{i_2+(r-e)\gamma,i_3+(r-e)\gamma\}} t_{\{i_3+(r-s)\gamma,i_4+(r-s)\gamma\}} t_{\{i_1+(r-u)\gamma,i_4+(r-u)\gamma\}}.\end{aligned} \tag{5.58}$$

Four of the seven cases are illustrated in Fig. 5.12.

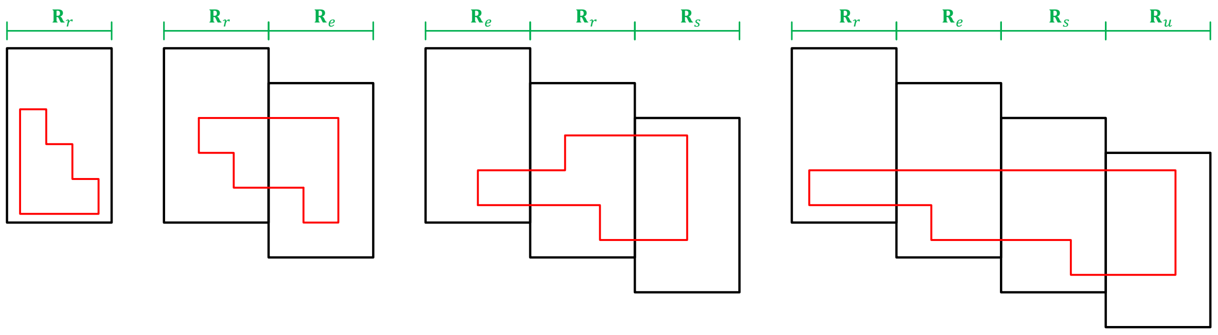


Figure 5.12: An instance of Pattern P_9 in Case 9.1, in Case 9.3, in Case 9.6, and in Case 9.7, from left to right. For simplicity, we have $e = r + y_1$, where $y_1 \in \{-1, 1\}$, $s = e + y_2$, where $y_2 \in \{1, 2\}$, and $u = s + 1$.

Theorem 16. *The total number of instances of Pattern P_9 in the binary protograph of an SC code that has parameters $\gamma \geqslant 3$, κ, m, $L \geqslant 2m + 1$, and $\mathcal{O}$, is*

$$F_{P_9} = \sum_{k=1}^{2m+1} (L - k + 1) F^k_{P_9,1}, \tag{5.59}$$

where $F^k_{P_9,1}$, $k \in \{1, 2, \ldots, 2m + 1\}$, are given by:

$$\begin{aligned}
F^1_{P_9,1} = {} & \frac{1}{2} \sum_{\{i_1,i_3\}\subset\{0,\ldots,(m+1)\gamma-1\},\{i_2,i_4\}\subset\{0,\ldots,(m+1)\gamma-1\}} \mathcal{A}_{P_9}\Big(t_{\{i_1,i_2\}}, t_{\{i_2,i_3\}}, t_{\{i_3,i_4\}}, t_{\{i_1,i_4\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_1,i_2,i_4\}}, t_{\{i_1,i_3,i_4\}} \\
& \qquad , t_{\{i_2,i_3,i_4\}}, t_{\{i_1,i_2,i_3,i_4\}}\Big), \\
F^2_{P_9,1} = {} & \sum_{\{i_1,i_4\}\subset\{\gamma,\ldots,(m+1)\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_3\in\{0,\ldots,(m+1)\gamma-1\}} \mathcal{B}_{P_9}\Big(t_{\{i_1,i_2\}}, t_{\{i_2,i_3\}}, t_{\{i_3,i_4\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_2,i_3,i_4\}}, t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_1-\gamma,i_4-\gamma\}}\Big) \\
& + \sum_{\{i_1,i_4\}\subset\{0,\ldots,m\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_3\in\{0,\ldots,(m+1)\gamma-1\}} \mathcal{B}_{P_9}\Big(t_{\{i_1,i_2\}}, t_{\{i_2,i_3\}}, t_{\{i_3,i_4\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_2,i_3,i_4\}}, t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_1+\gamma,i_4+\gamma\}}\Big) \\
& + \sum_{\{i_1,i_3\}\subset\{\gamma,\ldots,(m+1)\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_4\in\{\gamma,\ldots,(m+2)\gamma-1\}} \mathcal{C}_{P_9}\Big(t_{\{i_1,i_2\}}, t_{\{i_2,i_3\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_3-\gamma,i_4-\gamma\}}, t_{\{i_1-\gamma,i_4-\gamma\}}, t_{\{i_1-\gamma,i_3-\gamma,i_4-\gamma\}}\Big) \\
& + \frac{1}{2} \sum_{\{i_1,i_4\}\subset\{\gamma,\ldots,(m+1)\gamma-1\},i_2\in\{\gamma,\ldots,(m+1)\gamma-1\},i_3\in\{\gamma,\ldots,(m+1)\gamma-1\}} \mathcal{D}_{P_9}\Big(t_{\{i_1,i_2\}}, t_{\{i_3,i_4\}}, t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_2-\gamma,i_3-\gamma\}}, t_{\{i_1-\gamma,i_4-\gamma\}}, t_{\{i_1-\gamma,i_2-\gamma,i_3-\gamma,i_4-\gamma\}}\Big), \\
F^3_{P_9,1} = {} & \sum_{\{i_1,i_4\}\subset\{2\gamma,\ldots,(m+1)\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_3\in\{0,\ldots,(m+1)\gamma-1\}} \mathcal{B}_{P_9}\Big(t_{\{i_1,i_2\}}, t_{\{i_2,i_3\}}, t_{\{i_3,i_4\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_2,i_3,i_4\}}, t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_1-2\gamma,i_4-2\gamma\}}\Big) \\
& + \sum_{\{i_1,i_4\}\subset\{0,\ldots,(m-1)\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_3\in\{0,\ldots,(m+1)\gamma-1\}} \mathcal{B}_{P_9}\Big(t_{\{i_1,i_2\}}, t_{\{i_2,i_3\}}, t_{\{i_3,i_4\}}, t_{\{i_1,i_2,i_3\}}, t_{\{i_2,i_3,i_4\}}, t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_1+2\gamma,i_4+2\gamma\}}\Big)
\end{aligned}$$

$$
\begin{aligned}
&+ \sum_{\{i_1,i_3\}\subset\{2\gamma,\ldots,(m+1)\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_4\in\{2\gamma,\ldots,(m+3)\gamma-1\}} \mathcal{C}_{P_9}\Big(t_{\{i_1,i_2\}},t_{\{i_2,i_3\}},t_{\{i_1,i_2,i_3\}},t_{\{i_3-2\gamma,i_4-2\gamma\}},t_{\{i_1-2\gamma,i_4-2\gamma\}},t_{\{i_1-2\gamma,i_3-2\gamma,i_4-2\gamma\}}\Big)\\
&+ \frac{1}{2}\sum_{\{i_1,i_4\}\subset\{2\gamma,\ldots,(m+1)\gamma-1\},i_2\in\{2\gamma,\ldots,(m+1)\gamma-1\},i_3\in\{2\gamma,\ldots,(m+1)\gamma-1\}} \mathcal{D}_{P_9}\Big(t_{\{i_1,i_2\}},t_{\{i_3,i_4\}},t_{\{i_1,i_2,i_3,i_4\}},t_{\{i_2-2\gamma,i_3-2\gamma\}},t_{\{i_1-2\gamma,i_4-2\gamma\}}\\
&\qquad ,t_{\{i_1-2\gamma,i_2-2\gamma,i_3-2\gamma,i_4-2\gamma\}}\Big)\\
&+ \sum_{i_1\in\{2\gamma,\ldots,(m+1)\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_3\in\{\gamma,\ldots,(m+1)\gamma-1\},i_4\in\{2\gamma,\ldots,(m+2)\gamma-1\}} \mathcal{E}_{P_9}\Big(t_{\{i_1,i_2\}},t_{\{i_2,i_3\}},t_{\{i_1,i_2,i_3\}},t_{\{i_3-\gamma,i_4-\gamma\}},t_{\{i_1-2\gamma,i_4-2\gamma\}}\Big)\\
&+ \sum_{i_1\in\{\gamma,\ldots,(m+1)\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_3\in\{0,\ldots,m\gamma-1\},i_4\in\{\gamma,\ldots,m\gamma-1\}} \mathcal{E}_{P_9}\Big(t_{\{i_1,i_2\}},t_{\{i_2,i_3\}},t_{\{i_1,i_2,i_3\}},t_{\{i_3+\gamma,i_4+\gamma\}},t_{\{i_1-\gamma,i_4-\gamma\}}\Big)\\
&+ \sum_{i_1\in\{0,\ldots,m\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_3\in\{0,\ldots,(m-1)\gamma-1\},i_4\in\{-\gamma,\ldots,(m-1)\gamma-1\}} \mathcal{E}_{P_9}\Big(t_{\{i_1,i_2\}},t_{\{i_2,i_3\}},t_{\{i_1,i_2,i_3\}},t_{\{i_3+2\gamma,i_4+2\gamma\}},t_{\{i_1+\gamma,i_4+\gamma\}}\Big)\\
&+ \sum_{\{i_1,i_4\}\subset\{2\gamma,\ldots,(m+1)\gamma-1\},i_2\in\{\gamma,\ldots,(m+1)\gamma-1\},i_3\in\{\gamma,\ldots,(m+1)\gamma-1\}} \mathcal{G}_{P_9}\Big(t_{\{i_1,i_2\}},t_{\{i_3,i_4\}},t_{\{i_1,i_2,i_3,i_4\}},t_{\{i_2-\gamma,i_3-\gamma\}},t_{\{i_1-2\gamma,i_4-2\gamma\}}\Big)\\
&+ \sum_{\{i_1,i_4\}\subset\{\gamma,\ldots,(m+1)\gamma-1\},i_2\in\{0,\ldots,m\gamma-1\},i_3\in\{0,\ldots,m\gamma-1\}} \mathcal{G}_{P_9}\Big(t_{\{i_1,i_2\}},t_{\{i_3,i_4\}},t_{\{i_1,i_2,i_3,i_4\}},t_{\{i_2+\gamma,i_3+\gamma\}},t_{\{i_1-\gamma,i_4-\gamma\}}\Big)\\
&+ \sum_{\{i_1,i_4\}\subset\{0,\ldots,m\gamma-1\},i_2\in\{0,\ldots,(m-1)\gamma-1\},i_3\in\{0,\ldots,(m-1)\gamma-1\}} \mathcal{G}_{P_9}\Big(t_{\{i_1,i_2\}},t_{\{i_3,i_4\}},t_{\{i_1,i_2,i_3,i_4\}},t_{\{i_2+2\gamma,i_3+2\gamma\}},t_{\{i_1+\gamma,i_4+\gamma\}}\Big),
\end{aligned}
$$

$$
\begin{aligned}
F_{P_9,1}^{k\geqslant 4} &= \sum_{\{i_1,i_4\}\subset\{(k-1)\gamma,\ldots,(m+1)\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_3\in\{0,\ldots,(m+1)\gamma-1\}} \mathcal{B}_{P_9}\Big(t_{\{i_1,i_2\}},t_{\{i_2,i_3\}},t_{\{i_3,i_4\}},t_{\{i_1,i_2,i_3\}},t_{\{i_2,i_3,i_4\}},t_{\{i_1,i_2,i_3,i_4\}},t_{\{i_1+(1-k)\gamma,i_4+(1-k)\gamma\}}\Big)\\
&+ \sum_{\{i_1,i_4\}\subset\{0,\ldots,(m-k+2)\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_3\in\{0,\ldots,(m+1)\gamma-1\}} \mathcal{B}_{P_9}\Big(t_{\{i_1,i_2\}},t_{\{i_2,i_3\}},t_{\{i_3,i_4\}},t_{\{i_1,i_2,i_3\}},t_{\{i_2,i_3,i_4\}},t_{\{i_1,i_2,i_3,i_4\}},t_{\{i_1+(k-1)\gamma,i_4+(k-1)\gamma\}}\Big)\\
&+ \sum_{\{i_1,i_3\}\subset\{(k-1)\gamma,\ldots,(m+1)\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_4\in\{(k-1)\gamma,\ldots,(m+k)\gamma-1\}} \mathcal{C}_{P_9}\Big(t_{\{i_1,i_2\}},t_{\{i_2,i_3\}},t_{\{i_1,i_2,i_3\}},t_{\{i_3+(1-k)\gamma,i_4+(1-k)\gamma\}},t_{\{i_1+(1-k)\gamma,i_4+(1-k)\gamma\}}\\
&\qquad ,t_{\{i_1+(1-k)\gamma,i_3+(1-k)\gamma,i_4+(1-k)\gamma\}}\Big)\\
&+ \frac{1}{2}\sum_{\{i_1,i_4\}\subset\{(k-1)\gamma,\ldots,(m+1)\gamma-1\},i_2\in\{(k-1)\gamma,\ldots,(m+1)\gamma-1\},i_3\in\{(k-1)\gamma,\ldots,(m+1)\gamma-1\}} \mathcal{D}_{P_9}\Big(t_{\{i_1,i_2\}},t_{\{i_3,i_4\}},t_{\{i_1,i_2,i_3,i_4\}},t_{\{i_2+(1-k)\gamma,i_3+(1-k)\gamma\}},t_{\{i_1+(1-k)\gamma,i_4+(1-k)\gamma\}}\\
&\qquad ,t_{\{i_1+(1-k)\gamma,i_2+(1-k)\gamma,i_3+(1-k)\gamma,i_4+(1-k)\gamma\}}\Big)\\
&+ \sum_{h=2}^{k-1}\sum_{i_1\in\{(k-1)\gamma,\ldots,(m+1)\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_3\in\{(h-1)\gamma,\ldots,(m+1)\gamma-1\},i_4\in\{(k-1)\gamma,\ldots,(m+h)\gamma-1\}} \mathcal{E}_{P_9}\Big(t_{\{i_1,i_2\}},t_{\{i_2,i_3\}},t_{\{i_1,i_2,i_3\}},t_{\{i_3+(1-h)\gamma,i_4+(1-h)\gamma\}},t_{\{i_1+(1-k)\gamma,i_4+(1-k)\gamma\}}\Big)\\
&+ \sum_{h=2}^{k-1}\sum_{i_1\in\{(k-h)\gamma,\ldots,(m+1)\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_3\in\{0,\ldots,(m-h+2)\gamma-1\},i_4\in\{(k-h)\gamma,\ldots,(m-h+2)\gamma-1\}} \mathcal{E}_{P_9}\Big(t_{\{i_1,i_2\}},t_{\{i_2,i_3\}},t_{\{i_1,i_2,i_3\}},t_{\{i_3+(h-1)\gamma,i_4+(h-1)\gamma\}},t_{\{i_1+(h-k)\gamma,i_4+(h-k)\gamma\}}\Big)\\
&+ \sum_{h=2}^{k-1}\sum_{i_1\in\{0,\ldots,(m-k+h+1)\gamma-1\},i_2\in\{0,\ldots,(m+1)\gamma-1\},i_3\in\{0,\ldots,(m-k+2)\gamma-1\},i_4\in\{(h-k)\gamma,\ldots,(m-k+2)\gamma-1\}} \mathcal{E}_{P_9}\Big(t_{\{i_1,i_2\}},t_{\{i_2,i_3\}},t_{\{i_1,i_2,i_3\}},t_{\{i_3+(k-1)\gamma,i_4+(k-1)\gamma\}},t_{\{i_1+(k-h)\gamma,i_4+(k-h)\gamma\}}\Big)
\end{aligned}
$$

$$
\begin{aligned}
&+ \sum_{h=2}^{k-1} \sum_{\{i_1,i_4\}\subset\{(k-1)\gamma,\dots,(m+1)\gamma-1\}, i_2\in\{(h-1)\gamma,\dots,(m+1)\gamma-1\}, i_3\in\{(h-1)\gamma,\dots,(m+1)\gamma-1\}} \mathcal{G}_{P_9}\Big(t_{\{i_1,i_2\}}, t_{\{i_3,i_4\}}, t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_2+(1-h)\gamma,i_3+(1-h)\gamma\}}, t_{\{i_1+(1-k)\gamma,i_4+(1-k)\gamma\}}\Big)\\
&+ \sum_{h=2}^{k-1} \sum_{\{i_1,i_4\}\subset\{(k-h)\gamma,\dots,(m+1)\gamma-1\}, i_2\in\{0,\dots,(m-h+2)\gamma-1\}, i_3\in\{0,\dots,(m-h+2)\gamma-1\}} \mathcal{G}_{P_9}\Big(t_{\{i_1,i_2\}}, t_{\{i_3,i_4\}}, t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_2+(h-1)\gamma,i_3+(h-1)\gamma\}}, t_{\{i_1+(h-k)\gamma,i_4+(h-k)\gamma\}}\Big)\\
&+ \sum_{h=2}^{k-1} \sum_{\{i_1,i_4\}\subset\{0,\dots,(m-k+h+1)\gamma-1\}, i_2\in\{0,\dots,(m-k+2)\gamma-1\}, i_3\in\{0,\dots,(m-k+2)\gamma-1\}} \mathcal{G}_{P_9}\Big(t_{\{i_1,i_2\}}, t_{\{i_3,i_4\}}, t_{\{i_1,i_2,i_3,i_4\}}, t_{\{i_2+(k-1)\gamma,i_3+(k-1)\gamma\}}, t_{\{i_1+(k-h)\gamma,i_4+(k-h)\gamma\}}\Big)\\
+ &\sum_{h=2}^{k-2} \sum_{w=h+1}^{k-1} \sum_{\substack{i_1\in\{(k-1)\gamma,\dots,(m+1)\gamma-1\}, i_2\in\{(h-1)\gamma,\dots,(m+1)\gamma-1\},\\ i_3\in\{(w-1)\gamma,\dots,(m+h)\gamma-1\}, i_4\in\{(k-1)\gamma,\dots,(m+w)\gamma-1\}}} \mathcal{I}_{P_9}\Big(t_{\{i_1,i_2\}}, t_{\{i_2+(1-h)\gamma,i_3+(1-h)\gamma\}}, t_{\{i_3+(1-w)\gamma,i_4+(1-w)\gamma\}}, t_{\{i_1+(1-k)\gamma,i_4+(1-k)\gamma\}}\Big)\\
+ &\sum_{h=2}^{k-2} \sum_{w=h+1}^{k-1} \sum_{\substack{i_1\in\{(k-1)\gamma,\dots,(m+1)\gamma-1\}, i_2\in\{(w-1)\gamma,\dots,(m+1)\gamma-1\},\\ i_3\in\{(w-1)\gamma,\dots,(m+h)\gamma-1\}, i_4\in\{(k-1)\gamma,\dots,(m+h)\gamma-1\}}} \mathcal{I}_{P_9}\Big(t_{\{i_1,i_2\}}, t_{\{i_2+(1-w)\gamma,i_3+(1-w)\gamma\}}, t_{\{i_3+(1-h)\gamma,i_4+(1-h)\gamma\}}, t_{\{i_1+(1-k)\gamma,i_4+(1-k)\gamma\}}\Big)\\
+ &\sum_{h=2}^{k-2} \sum_{w=h+1}^{k-1} \sum_{\substack{i_1\in\{(w-1)\gamma,\dots,(m+1)\gamma-1\}, i_2\in\{(h-1)\gamma,\dots,(m+1)\gamma-1\},\\ i_3\in\{(k-1)\gamma,\dots,(m+h)\gamma-1\}, i_4\in\{(k-1)\gamma,\dots,(m+w)\gamma-1\}}} \mathcal{I}_{P_9}\Big(t_{\{i_1,i_2\}}, t_{\{i_2+(1-h)\gamma,i_3+(1-h)\gamma\}}, t_{\{i_3+(1-k)\gamma,i_4+(1-k)\gamma\}}, t_{\{i_1+(1-w)\gamma,i_4+(1-w)\gamma\}}\Big),
\end{aligned}
\tag{5.60}
$$

with $\overline{i_1} \neq \overline{i_2}$, $i_1 \neq i_3$, $\overline{i_1} \neq \overline{i_4}$, $\overline{i_2} \neq \overline{i_3}$, $i_2 \neq i_4$, *and* $\overline{i_3} \neq \overline{i_4}$.

After deriving the expressions of F_{P_ℓ}, $\forall \ell$, as functions of the overlap parameters in $\mathcal{O}$, we use (5.3), (5.4), and [37, Lemma 3] to express F_{sum} as a function of the parameters in $\mathcal{O}_{\text{ind}}$ (which is the set of all independent non-zero overlap parameters). Thus, our ***discrete optimization problem*** is:

$$F^*_{\text{sum}} = \min_{\mathcal{O}_{\text{ind}}} F_{\text{sum}}. \tag{5.61}$$

The constraints of the optimization problem in (5.61) are linear constraints capturing the interval constraints under which the resultant partitioning is valid. We also add the balanced partitioning constraint, which guarantees a balanced distribution of the non-zero circulants among the $(m + 1)$ component matrices. (see also [37] and [38]). A balanced partitioning is preferred in order to prevent the situation where a group of non-zero elements in a particular component matrix are involved in significantly more cycles than the remaining non-zero elements. This constraint, although it might result in a sub-optimal solution in the protograph (in a few cases), is observed to be very beneficial when we apply the CPO to construct the final code.

As with the set $\mathcal{O}_{\text{ind}}$, the optimization constraints depend only on code parameters, and not on the common substructure of interest (which depends on the channel). For example, in the case of $\gamma = 3$, $m = 1$, and any κ, $\mathcal{O}_{\text{ind}} = \{t_0, t_1, t_2, t_{\{0,1\}}, t_{\{0,2\}}, t_{\{1,2\}}, t_{\{0,1,2\}}\}$, and the optimization constraints are (see also [37] and [38]):

$$\begin{aligned}
&0 \leqslant t_0 \leqslant \kappa, \qquad 0 \leqslant t_{\{0,1\}} \leqslant t_0, \qquad t_{\{0,1\}} \leqslant t_1 \leqslant \kappa - t_0 + t_{\{0,1\}}, \\
&0 \leqslant t_{\{0,1,2\}} \leqslant t_{\{0,1\}}, \qquad\qquad t_{\{0,1,2\}} \leqslant t_{\{0,2\}} \leqslant t_0 - t_{\{0,1\}} + t_{\{0,1,2\}}, \\
&t_{\{0,1,2\}} \leqslant t_{\{1,2\}} \leqslant t_1 - t_{\{0,1\}} + t_{\{0,1,2\}}, \\
&t_{\{0,2\}} + t_{\{1,2\}} - t_{\{0,1,2\}} \leqslant t_2 \leqslant \kappa - t_0 - t_1 + t_{\{0,1\}} + t_{\{0,2\}} + t_{\{1,2\}} - t_{\{0,1,2\}}, \\
&\textit{and} \ \lfloor 3\kappa/2 \rfloor \leqslant t_0 + t_1 + t_2 \leqslant \lceil 3\kappa/2 \rceil .
\end{aligned} \tag{5.62}$$

The solution of this optimization problem is not unique. However, since all the solutions have the same performance (e.g., they all achieve F^*_{sum}, see also [38]), we work with one of these solutions, and call it an optimal vector, $\mathbf{t}^*$.

5.5 CPO: Customization for PR Systems

Using the optimal vector $\mathbf{t}^*$, computed as described in the previous section, $\mathbf{H}^{\text{p}}$ is partitioned and the protograph matrix of the SC code, $\mathbf{H}^{\text{p}}_{\text{SC}}$, is constructed. The next step is preventing as many objects in the protograph as possible from being reflected in the unlabeled graph of the SC code, via optimizing the circulant powers using the CPO. Here, the CPO is customized for the $(4, 4(\gamma-2))$ object, which is the common substructure for detrimental configurations in the case of PR systems (see also Fig. 5.2).

From the previous analysis, a Pattern P_ℓ spans at most either $m+1$ or $2m+1$ consecutive replicas, depending on the value of ℓ. Let $\xi = 2m+1$. Thus, in the CPO, it suffices to operate on the PM $\mathbf{\Pi}^{\xi,\text{p}}_1$, which is the non-zero part of the first ξ replicas in $\mathbf{H}^{\text{p}}_{\text{SC}}$, and has the size $(\xi+m)\gamma \times \xi\kappa$. The circulant powers associated with the 1's in $\mathbf{H}^{\text{p}}$ are defined as $f_{i,j}$, where

$0 \leqslant i \leqslant \gamma - 1$ and $0 \leqslant j \leqslant \kappa - 1$. Let the circulant powers associated with the 1's in $\mathbf{\Pi}_1^{\xi,\mathrm{p}}$ be $f'_{i',j'}$, where $0 \leqslant i' \leqslant (\xi + m)\gamma - 1$ and $0 \leqslant j' \leqslant \xi\kappa - 1$. From the repetitive nature of the PM $\mathbf{\Pi}_1^{\xi,\mathrm{p}}$, $f'_{i',j'} = f_{\overline{i'},\widetilde{j'}}$, where $\overline{i'} = (i' \bmod \gamma)$ and $\widetilde{j'} = (j' \bmod \kappa)$. Define our cycle-8 candidate in the graph of $\mathbf{\Pi}_1^{\xi,\mathrm{p}}$ as $c_1 - v_1 - c_2 - v_2 - c_3 - v_3 - c_4 - v_4$, which is a particular way of traversing a pattern and not necessarily a protograph cycle (see also Figures 5.2 and 5.3). This candidate results in z (or $z/2$ in the case of P_1 only) cycles of length 8 after lifting if and only if [87]:

$$f'_{c_1,v_1} + f'_{c_2,v_2} + f'_{c_3,v_3} + f'_{c_4,v_4} \equiv f'_{c_1,v_2} + f'_{c_2,v_3} + f'_{c_3,v_4} + f'_{c_4,v_1} \pmod{z}. \tag{5.63}$$

The goal is to prevent as many cycle-8 candidates in the graph of $\mathbf{H}_{\mathrm{SC}}^{\mathrm{p}}$ as possible from being converted into z (or $z/2$ in the case of P_1) $(4, 4(\gamma-2))$ UASs/UTSs in the graph of $\mathbf{H}_{\mathrm{SC}}$, which is the unlabeled graph of the SC code. In other words, a cycle-8 candidate in the graph of $\mathbf{H}_{\mathrm{SC}}^{\mathrm{p}}$ is allowed to be converted into multiple $(4, 4(\gamma-2) - 2\delta)$ UASs/UTSs, with $\delta \in \{1, 2\}$, as long as they are not $(4, 0)$ UASs, in the unlabeled graph since these are not instances of the common substructure of interest. These $(4, 4(\gamma-2) - 2\delta)$ UASs/UTSs, $\delta \in \{1, 2\}$, are cycles of length 8 with ***internal connections***, which means v_1 and v_3 are adjacent or/and v_2 and v_4 are adjacent (see Fig. 5.2). For the cycle-8 candidate in the graph of $\mathbf{\Pi}_1^{\xi,\mathrm{p}}$ that is described in the previous paragraph and has a CN, say c_5, connecting v_1 and v_3, in order to have this internal connection in the lifted cycles, the following condition for a cycle of length 6 must be satisfied in addition to (5.63):

$$f'_{c_1,v_1} + f'_{c_2,v_2} + f'_{c_5,v_3} \equiv f'_{c_1,v_2} + f'_{c_2,v_3} + f'_{c_5,v_1} \pmod{z}. \tag{5.64}$$

Similarly, for that cycle-8 candidate in the graph of $\mathbf{\Pi}_1^{\xi,\mathrm{p}}$ that has a CN, say c_6, connecting v_2 and v_4, in order to have this internal connection in the lifted cycles, the following condition for a cycle of length 6 must be satisfied in addition to (5.63):

$$f'_{c_1,v_1} + f'_{c_6,v_2} + f'_{c_4,v_4} \equiv f'_{c_1,v_2} + f'_{c_6,v_4} + f'_{c_4,v_1} \pmod{z}. \tag{5.65}$$

Note that the two CNs, c_5 and c_6, have to be different from the CNs of the pattern itself in order that we consider them in the CPO algorithm as internal connections. The reason is that the final unlabeled graphs of our codes must have no cycles of length 4 (which is also why (5.63) is applied for P_1 since $f'_{c_1,v_1} + f'_{c_2,v_2} \equiv f'_{c_1,v_2} + f'_{c_2,v_1} \pmod{z}$ is not allowed for any protograph cycle of length 4, $c_1 - v_1 - c_2 - v_2$).

The following lemma discusses the internal connections for different patterns in the protograph.

Lemma 25. *Let η_{P_ℓ} be the maximum number of internal connections Pattern P_ℓ can have (multiple internal connections between the same two VNs are only counted once). Then,*

$$\eta_{P_\ell} = \begin{cases} 0, & \ell \in \{1,3,5\}, \\ 1, & \ell \in \{2,6,8\}, \\ 2, & \ell \in \{4,7,9\}. \end{cases} \tag{5.66}$$

Proof. A protograph pattern, P_ℓ, with only two VNs ($\ell \in \{1,3,5\}$) cannot have any internal connections. A protograph pattern with three VNs ($\ell \in \{2,6,8\}$) can have at most one internal connection. A protograph pattern with four VNs ($\ell \in \{4,7,9\}$) can have up to two internal connection, which completes the proof. □

The case of multiple internal connections between the same two VNs is addressed in the CPO algorithm.

The steps of the customized CPO algorithm for SC codes that have parameters $\gamma \geqslant 3$, κ, m, and $L \geqslant 2m+1$, are:

1. Assign initial circulant powers to all the $\gamma\kappa$ 1's in $\mathbf{H}^{\text{p}}$. In this work, our initial powers are as in SCB codes. For example, $f_{i,j} = (i^2)(2j)$, $0 \leqslant i \leqslant \gamma - 1$ and $0 \leqslant j \leqslant \kappa - 1$ (initially, no cycles of length 4 are in $\mathbf{H}_{\text{SC}}$).

2. Construct $\mathbf{\Pi}_1^{\xi,\mathrm{p}}$ via $\mathbf{H}^{\mathrm{p}}$ and $\mathbf{t}^*$. Circulant powers of the 1's in $\mathbf{\Pi}_1^{\xi,\mathrm{p}}$, $f'_{i',j'}$, are obtained from the 1's in $\mathbf{H}^{\mathrm{p}}$.

3. Define a counting variable $\psi_{i,j}$, $0 \leqslant i \leqslant \gamma - 1$ and $0 \leqslant j \leqslant \kappa - 1$, for each of the 1's in $\mathbf{H}^{\mathrm{p}}$. Define another counting variable $\psi'_{i',j'}$, $0 \leqslant i' \leqslant (\xi + m)\gamma - 1$ and $0 \leqslant j' \leqslant \xi\kappa - 1$, for each of the elements in $\mathbf{\Pi}_1^{\xi,\mathrm{p}}$. Initialize all the variables in this step with zeros. Only $\xi\gamma\kappa$ counting variables of the form $\psi'_{i',j'}$ are associated with 1's in $\mathbf{\Pi}_1^{\xi,\mathrm{p}}$. The other variables remain zeros.

4. Locate all instances of the nine patterns in $\mathbf{\Pi}_1^{\xi,\mathrm{p}}$. Note that locating P_1 means also locating all cycles of length 4 in $\mathbf{\Pi}_1^{\xi,\mathrm{p}}$, which is needed.

5. Determine the ζ_{P_ℓ} ways to traverse each instance of P_ℓ, $\forall \ell$, to reach $(4, 4(\gamma - 2))$ UASs/UTSs in the unlabeled graph, which are the ζ_{P_ℓ} cycle-8 candidates.

6. Specify all internal connections (CNs) in each candidate determined in Step 5 if they can exist.

7. For each cycle-8 candidate in $\mathbf{\Pi}_1^{\xi,\mathrm{p}}$, check whether (5.63) is satisfied for its circulant powers or not.

8. If (5.63) is satisfied, and the candidate has no internal connections, or (5.63) is satisfied and the candidate has internal connection(s) but neither (5.64) nor (5.65) is satisfied for any internal connection, mark this cycle-8 candidate as an ***active candidate***.

9. Let $F^{k,\mathrm{a}}_{P_\ell,1}$, where $k \in \{1, 2, \ldots, \xi\}$, be the number of active candidates of P_ℓ starting at the first replica and spanning k consecutive replicas in $\mathbf{\Pi}_1^{\xi,\mathrm{p}}$. Thus, the number of active candidates of P_ℓ spanning k consecutive replicas in $\mathbf{\Pi}_1^{\xi,\mathrm{p}}$ is $(\xi - k + 1)F^{k,\mathrm{a}}_{P_\ell,1}$. *(For example, for $k = 1$, $\xi F^{1,\mathrm{a}}_{P_\ell,1}$ is the number of active candidates of P_ℓ, for any value of ℓ, spanning one replica in $\mathbf{\Pi}_1^{\xi,\mathrm{p}}$.)*

10. Compute the number of $(4, 4(\gamma-2))$ UASs/UTSs in $\mathbf{H}_{\text{SC}}$ using the following formula (see also [37]):
$$F_{\text{SC}} = \sum_{\ell=1}^{9} \sum_{k=1}^{\xi} \left((L-k+1) F_{P_\ell,1}^{k,\text{a}} \right) z_{P_\ell}, \tag{5.67}$$
where $z_{P_\ell} = z/2$ if $\ell = 1$, and $z_{P_\ell} = z$ otherwise. Recall that $\xi = 2m+1$.

11. Count the number of active candidates each 1 in $\mathbf{\Pi}_1^{\xi,\text{p}}$ is involved in. Assign weight $w_k = (L-k+1)/(\xi-k+1)$ to the number of active candidates spanning k consecutive replicas in $\mathbf{\Pi}_1^{\xi,\text{p}}$ (see also [37]). Multiply w_k by $1/2$ if the candidate is associated to P_1. *(For example, for $k = \xi$, the weight of the number of active candidates spanning ξ consecutive replicas is $(L - \xi + 1)$.)*

12. Store the weighted count associated with each 1 in $\mathbf{\Pi}_1^{\xi,\text{p}}$, which is indexed by (i', j'), in $\psi'_{i',j'}$.

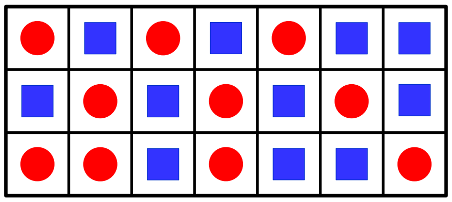
13. Calculate the counting variables $\psi_{i,j}$, $\forall i, j$, associated with the 1's in $\mathbf{H}^{\text{p}}$ from the counting variables $\psi'_{i',j'}$ associated with the 1's in $\mathbf{\Pi}_1^{\xi,\text{p}}$ (computed in Steps 11 and 12) using the following formula:
$$\psi_{i,j} = \sum_{i' : \overline{i'} = i} \sum_{\substack{j' : \widetilde{j'} = j \\ \mathbf{\Pi}_1^{\xi,\text{p}}[i'][j'] \neq 0}} \psi'_{i',j'}, \tag{5.68}$$

14. Sort these $\gamma\kappa$ 1's of $\mathbf{H}^{\text{p}}$ in a list descendingly according to the counts in $\psi_{i,j}$, $\forall i, j$.

15. Pick a subset of 1's from the top of this list, and change the circulant powers associated with them.

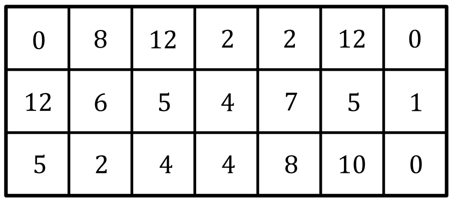
16. Using these interim powers, do Steps 7, 8, 9, and 10.

17. If F_{SC} is reduced while maintaining no cycles of length 4 and no $(4, 0)$ objects (in the case of $\gamma = 3$) in $\mathbf{H}_{\text{SC}}$, update F_{SC} and the circulant powers, then go to Step 11.



●	■	●	■	●	■	■
■	●	■	●	■	●	■
●	●	■	●	■	■	●

(a)



0	8	12	2	2	12	0
12	6	5	4	7	5	1
5	2	4	4	8	10	0

(b)

Figure 5.13: (a) The OO partitioning of $\mathbf{H}^{\mathrm{p}}$ (or $\mathbf{H}$) of the SC code in Example 24. Entries with circles (resp., squares) are assigned to $\mathbf{H}_0^{\mathrm{p}}$ (resp., $\mathbf{H}_1^{\mathrm{p}}$). (b) The circulant power arrangement for the circulants in $\mathbf{H}$.

18. Otherwise, return to Step 15 to pick a different set of circulant powers or/and a different subset of 1's (from the 1's in $\mathbf{H}^{\mathrm{p}}$).

19. Iterate until the target F_{SC} (set by the code designer) is achieved, or the reduction in F_{SC} approaches zero.

Step 15 in the CPO algorithm is performed heuristically. The number of 1's to work with depend on the circulant size, the values of the counts, and how these values are distributed. Moreover, tracking the counts of active candidates and the distribution of their values over different 1's in $\mathbf{H}^{\mathrm{p}}$ is the main factor to decide which 1's to select in each iteration.

Example 24. *Suppose we are designing an SC code with* $\gamma = 3$, $\kappa = 7$, $z = 13$, $m = 1$, *and* $L = 10$ *using the OO-CPO approach for PR systems. Solving the optimization problem in (5.61) gives an optimal vector* $\mathbf{t}^* = [t_0^* \; t_1^* \; t_2^* \; t_{\{0,1\}}^* \; t_{\{0,2\}}^* \; t_{\{1,2\}}^* \; t_{\{0,1,2\}}^*]^{\mathrm{T}} = [3 \; 3 \; 4 \; 0 \; 1 \; 2 \; 0]^{\mathrm{T}}$, *with* $F_{\mathrm{sum}}^* = 5{,}170$ *patterns (rounded weighted sum) in the graph of* $\mathbf{H}_{\mathrm{SC}}^{\mathrm{p}}$. *Fig. 5.13(a) shows how the partitioning is applied on* $\mathbf{H}^{\mathrm{p}}$ *(or* $\mathbf{H}$*). Next, applying the CPO results in* $2{,}613$ $(4,4)$ *UASs in the graph of* $\mathbf{H}_{\mathrm{SC}}$. *Fig. 5.13(b) shows the final circulant power arrangement for all circulants in* $\mathbf{H}$.

Remark 24. *After introducing the concept of patterns in this work, the OO-CPO approach can be easily extended to target other common substructures if needed.*

5.6 Experimental Results

In this section, we propose experimental results demonstrating the effectiveness of the OO-CPO approach compared with other code design techniques in PR (1-D MR) systems.

Remark 25. *In this section, all the codes used have no cycles of length* 4*. Moreover, we opted to work with circulant sizes $z > \kappa$ in order to give more freedom to the CPO, which results in less detrimental objects.*

First, we compare the total number of instances of the common substructure of interest in the unlabeled graphs of SC codes designed using various techniques. We present results for two groups of codes.

All the codes in the first group have $\gamma = 3$ (i.e., the common substructure of interest is the $(4, 4)$ UAS in Fig. 5.2) and $m \in \{1, 2\}$. We also choose $L = 10$ for this group. In addition to the uncoupled setting ($\mathbf{H}_0 = \mathbf{H}$ and $\mathbf{H}_1 = \mathbf{0}$), we show results for the following five SC code design techniques:

1. The CV technique (see [32]) with $m = 1$.
2. The OO technique with no CPO applied and with $m = 1$.
3. The OO technique with circulant powers optimized via the CPO (the OO-CPO approach) and with $m = 1$.
4. The OO technique with no CPO applied and with $m = 2$.
5. The OO technique with circulant powers optimized via the CPO (the OO-CPO approach) and with $m = 2$.

In the uncoupled setting in addition to the first, second, and fourth techniques, circulant powers as in SCB codes, $f_{i,j} = f(i)f(j) = (i^2)(2j)$, are used.

The results of the first group of codes for different choices of κ and z are listed in Table 5.1. For a particular choice of κ, z, m, and L, SC codes designed using these different techniques

Table 5.1: Number of $(4, 4)$ UASs in SC codes with $\gamma = 3$, $m \in \{1, 2\}$, and $L = 10$ designed using different techniques.

Design technique	Number of $(4,4)$ UASs			
	$\kappa = 7$, $z = 13$	$\kappa = 11$, $z = 23$	$\kappa = 13$, $z = 29$	$\kappa = 17$, $z = 37$
Uncoupled with SCB	32,370	254,610	540,850	1,700,890
SC CV with SCB and $m = 1$	9,464	91,333	197,084	652,347
SC OO with SCB and $m = 1$	6,500	53,130	123,395	440,818
SC OO-CPO and $m = 1$	2,613	32,361	70,151	254,005
SC OO with SCB and $m = 2$	3,172	27,508	60,233	194,176
SC OO-CPO and $m = 2$	819	13,110	32,074	117,697

all have block length $= \kappa z L$ and rate $\approx \left[1 - \frac{3(L+m)}{\kappa L}\right]$. Table 5.1 demonstrates the significant gains achieved by the OO-CPO approach compared with other techniques. In particular, for $m = 1$, the proposed OO-CPO approach achieves a reduction in the number of $(4, 4)$ UASs that ranges between 85% and 92% compared with the uncoupled setting, and between 61% and 72% compared with the CV technique. The table also illustrates the positive effect of increasing the memory of the SC code. In particular, the OO-CPO approach with $m = 2$ achieves a reduction in the number of $(4, 4)$ UASs that ranges between 54% and 69% compared with the OO-CPO approach with $m = 1$. Moreover, the importance of the two stages (the OO and the CPO) is highlighted by the numbers in Table 5.1.

As for the second group, all the codes have $\gamma = 4$ (i.e., the common substructure of interest is the $(4, 8)$ UTS in Fig. 5.2) and $m = 1$. We also choose $L = 10$ for this group. In addition to the uncoupled setting ($\mathbf{H}_0 = \mathbf{H}$ and $\mathbf{H}_1 = \mathbf{0}$), we show results for the following three SC code design techniques:

1. The CV technique (see [32]).

2. The OO technique with no CPO applied.

3. The OO technique with circulant powers optimized via the CPO (the OO-CPO approach).

In the uncoupled setting in addition to the first and second techniques, circulant powers as in SCB codes, $f_{i,j} = f(i)f(j) = (i^2)(2j)$, are used.

Table 5.2: Number of $(4, 8)$ UTSs in SC codes with $\gamma = 4$, $m = 1$, and $L = 10$ designed using different techniques.

Design technique	Number of $(4,8)$ UTSs			
	$\kappa = 7$, $z = 13$	$\kappa = 11$, $z = 23$	$\kappa = 13$, $z = 29$	$\kappa = 17$, $z = 37$
Uncoupled with SCB	131,820	1,034,310	2,193,850	7,081,430
SC CV with SCB	48,074	396,474	843,233	2,782,844
SC OO with SCB	27,729	230,230	508,544	1,667,886
SC OO-CPO	17,095	165,071	366,212	1,253,745

The results of the second group of codes for different choices of κ and z are listed in Table 5.2. For a particular choice of κ, z, and L, SC codes designed using these different techniques all have block length $= \kappa z L$ and rate $\approx \left[1 - \frac{4(L+1)}{\kappa L}\right]$. Table 5.2 again demonstrates the significant gains achieved by the OO-CPO approach compared with other techniques. In particular, the proposed OO-CPO approach achieves a reduction in the number of $(4, 8)$ UTSs that ranges between 82% and 87% compared with the uncoupled setting, and between 55% and 64% compared with the CV technique. Moreover, the importance of the two stages (the OO and the CPO) is again highlighted by the numbers in Table 5.2.

Second, we present simulation results of SC codes designed using various techniques over the PR channel. We use the PR channel described in Chapter 2. This channel incorporates inter-symbol interference (intrinsic memory), jitter, and electronic noise. The normalized channel density [50, 51] we use is 1.4, and the PR equalization target is [8 14 2]. The receiver consists of filtering units followed by a Bahl Cocke Jelinek Raviv (BCJR) detector [53], which is based on pattern-dependent noise prediction (PDNP) [54], in addition to a fast Fourier transform based q-ary sum-product algorithm (FFT-QSPA) LDPC decoder [52]. The number of global (detector-decoder) iterations is 10, and the number of local (decoder only) iterations is 20. Unless a codeword is reached, the decoder performs its prescribed number of local iterations for each global iteration.

In the simulations, we use five different codes. All the codes are defined over GF(4). Codes 5.1, 5.2, 5.3, and 5.4 have $\gamma = 3$, $\kappa = 19$, $z = 46$, $m = 1$, and $L = 5$. Thus, these codes have block length = 8,740 bits, and the SC codes have rate ≈ 0.81. Code 5.1 is

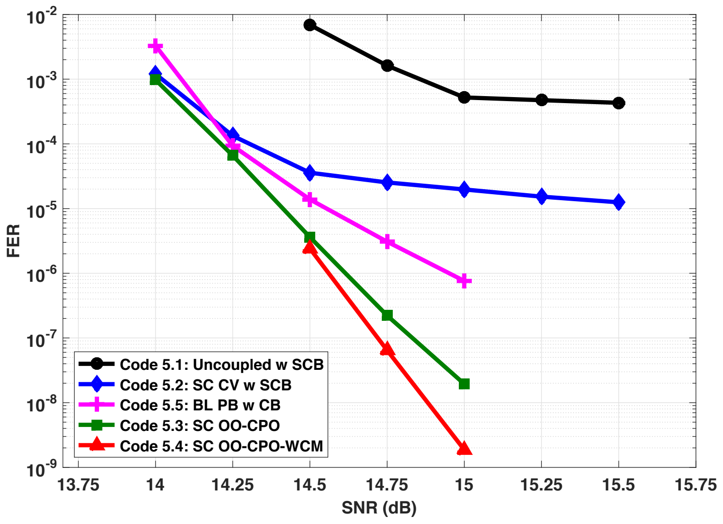


Figure 5.14: Simulation results over the PR channel for SC codes designed using different techniques and a BL code.

uncoupled. Code 5.2 is an SC code designed using the CV technique for PR channels as described in [32]. Codes 5.1 and 5.2 have SCB circulant powers of the form $f_{i,j} = (i^2)(2j)$. Code 5.3 is an SC code designed using the OO-CPO approach. Codes 5.1, 5.2, and 5.3 have unoptimized edge weights. Code 5.4 is the result of applying the WCM framework to Code 5.3 in order to optimize its edge weights. The numbers of $(4, 4)$ UASs in the unlabeled graphs of Codes 5.1, 5.2, and 5.3 are 2,425,120, 845,434, and 184,667, respectively. Code 5.5 is a block (BL) code, which is also protograph-based (PB), designed as in Chapter 3 and Chapter 4. Code 5.5 has column weight $= 3$, circulant size $= 46$, block length $= 8{,}832$ bits, rate ≈ 0.81 (same as all SC codes), and unoptimized weights (similar to all codes except Code 5.4). Since our main focus in this work is the performance, a relatively small value of L (which is 5) along with block decoding are used for SC Codes 5.1, 5.2, 5.3, and 5.4.

Fig. 5.14 demonstrates the effectiveness of the proposed OO-CPO approach in designing high performance SC codes for PR channels. In particular, Code 5.3 (designed using the OO-CPO approach) outperforms Code 5.2 (designed using the CV technique) by about 3 orders of magnitude at signal-to-noise ratio (SNR) $= 15$ dB, and by about 1.1 dB at FER $\approx 10^{-5}$. More intriguingly, Code 5.3 outperforms Code 5.5 (the block code) by about 1.6 orders of magnitude at SNR $= 15$ dB, and by almost 0.4 dB at FER $\approx 10^{-6}$. The performance of

Code 5.3 is better than the performance of Code 5.5 not only in the error floor region, but also in the waterfall region. An interesting observation is that, in the error profile of Code 5.3, we found no codewords of weights $\in \{6, 8\}$ (which are $(6, 0, 0, 9, 0)$ and $(8, 0, 0, 12, 0)$ BASTs) despite the dominant presence of such low weight codewords in the error profiles of Codes 5.1, 5.2, and 5.5 (see also Chapter 2 and Chapter 4 in addition to [40], [43], and [32]). From Fig. 5.14, the WCM framework achieves 1 order of magnitude additional gain.

An important reason behind the improved waterfall performance of Code 5.3 is the significant reduction in the multiplicity of low weight codewords achieved by the OO-CPO approach. This reduction is a result of the fact that such low weight codewords also have the $(4, 4)$ UAS as a common substructure in their configurations (see Fig. 5.2).

5.7 Concluding Remarks

We proposed the OO-CPO approach to optimally design binary and non-binary SC codes for PR channels, via minimizing the number of detrimental objects in the graph of the code. SC codes designed using the OO-CPO approach were shown to significantly outperform SC codes designed using techniques from the literature. More importantly, SC codes designed using our approach were demonstrated to outperform structured block codes with the same parameters.

Acknowledgement

The majority of the material in this chapter is in [45], which will be submitted to a journal soon. The work will also be presented in part at GLOBECOM 2018 [44]. The author would like to thank the collaborators in these publications.

5.8 Appendix

5.8.1 Proofs of Pattern P_1

Proof of Lemma 16

Proof. In Case 1.1, the number we are after is the number of ways to choose 2 overlaps out of $t_{\{i_1,i_2\}}$ overlaps, which is given by (5.6). In Case 1.2, the number we are after is the number of ways to choose 1 overlap out of $t_{\{i_1,i_2\}}$ and 1 overlap out of $t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma\}}$, which is given by (5.7). □

Proof of Theorem 8

Proof. To compute F_{P_1}, we use the formula in [37, Theorem 1], with χ, which is the maximum number of replicas the pattern can span, equals $m+1$. Since the overlaps of P_1 can exist in up to 2 replicas, we need to find expressions only for $F^{1}_{P_1,1}$ (overlaps are in 1 replica) and $F^{k\geqslant 2}_{P_1,1}$ (overlaps are in 2 replicas).

Then, $F^{1}_{P_1,1}$ is the sum of function $\mathcal{A}_{P_1}$ in (5.6), with $r=1$, over all possible values of $\{i_1,i_2\}$. Here, $\{i_1,i_2\}$ can take any distinct two values in the range from the start to the end of $\mathbf{R}_1$, i.e., from 0 to $(m+1)\gamma-1$ (see Fig. 5.4).

Moreover, $F^{k\geqslant 2}_{P_1,1}$ is the sum of function $\mathcal{B}_{P_1}$ in (5.7), with $r=1$ and $e=k$, over all possible values of $\{i_1,i_2\}$. Here, $\{i_1,i_2\}$ can take any distinct two values in the range from the start of $\mathbf{R}_k$ to the end of $\mathbf{R}_1$, i.e., from $(k-1)\gamma$ to $(m+1)\gamma-1$ (see also Fig. 5.4). □

5.8.2 Proofs of Pattern P_2

Proof of Lemma 17

Proof. In Case 2.1, the number we are after is the number of ways to choose 3 overlaps out of $t_{\{i_1,i_2\}}$ overlaps, which is given by (5.10). In Case 2.2, the number we are after is the number of ways to choose 2 overlap out of $t_{\{i_1,i_2\}}$ and 1 overlap out of $t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma\}}$, which is given by (5.11). In Case 2.3, the number we are after is the number of ways to choose 1 overlap out of $t_{\{i_1,i_2\}}$, 1 overlap out of $t_{\{i_1+(r-e)\gamma, i_2+(r-e)\gamma\}}$, and 1 overlap out of $t_{\{i_1+(r-s)\gamma, i_2+(r-s)\gamma\}}$, which is given by (5.12). $\square$

Proof of Theorem 9

Proof. To compute F_{P_2}, we use the formula in [37, Theorem 1], with $\chi = m+1$. Since the overlaps of P_2 can exist in up to 3 replicas, we need to find expressions only for $F^1_{P_2,1}$, $F^2_{P_2,1}$, and $F^{k\geqslant 3}_{P_2,1}$.

Then, $F^1_{P_2,1}$ is the sum of function $\mathcal{A}_{P_2}$ in (5.10), with $r = 1$, over all possible values of $\{i_1, i_2\}$. Here, $\{i_1, i_2\}$ can take any distinct two values in the range from the start to the end of $\mathbf{R}_1$, i.e., from 0 to $(m+1)\gamma - 1$ (see Fig. 5.5).

Regarding $F^2_{P_2,1}$, we need to distinguish between two situations; when $r < e$ (i.e., replica $\mathbf{R}_r$, which has two overlaps, comes before replica $\mathbf{R}_e$), and when $r > e$ (i.e., replica $\mathbf{R}_r$ comes after replica $\mathbf{R}_e$). This distinction gives the two summations of function $\mathcal{B}_{P_2}$ in $F^2_{P_2,1}$. For the first summation, $\mathcal{B}_{P_2}$ in (5.11) has $r = 1$ and $e = 2$. Thus, $\{i_1, i_2\}$ can take any distinct two values in the range from the start of $\mathbf{R}_2$ to the end of $\mathbf{R}_1$, i.e., from γ to $(m+1)\gamma - 1$ (see Fig. 5.5 for more illustration). For the second summation, $\mathcal{B}_{P_2}$ in (5.11) has $r = 2$ and $e = 1$. Thus, $\{i_1, i_2\}$ can take any distinct two values in the range from the start of $\mathbf{R}_2$ (which is now $\mathbf{R}_r$) to the end of $\mathbf{R}_1$, i.e., from 0 to $m\gamma - 1$.

As for $F^{k\geqslant 3}_{P_2,1}$, the overlaps can be in 2 replicas (the first two summations in $F^{k\geqslant 3}_{P_2,1}$) or 3 replicas (the third summation in $F^{k\geqslant 3}_{P_2,1}$). The first two summations are derived in a way similar to what we did for $F^2_{P_2,1}$, with a change in the summation indices; $\mathbf{R}_2$ is replaced by $\mathbf{R}_k$ here. For the third (double) summation, $\mathcal{C}_{P_2}$ in (5.12) has $r = 1$, $e = h$, and $s = k$. Thus, $\{i_1, i_2\}$ can take any distinct two values in the range from the start of $\mathbf{R}_k$ to the end of $\mathbf{R}_1$, i.e., from $(k-1)\gamma$ to $(m+1)\gamma - 1$ (see Fig. 5.5). The outer summation is over all possible values of h, and we have $1 < h < k$. $\square$

5.8.3 Proofs of Pattern P_3

Proof of Lemma 18

Proof. In Case 3.1, the number we are after is the number of ways to choose 2 overlaps out of $t_{\{i_1,i_2,i_3\}}$, which is given by (5.15). In Case 3.2, the number we are after is the number of ways to choose 1 overlap out of $t_{\{i_1,i_2,i_3\}}$ and 1 overlap out of $t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma,i_3+(r-e)\gamma\}}$, which is given by (5.16). $\square$

Proof of Theorem 10

Proof. To compute F_{P_3}, we use the formula in [37, Theorem 1], with $\chi = m+1$. Since the overlaps of P_3 can exist in up to 2 replicas, we need to find expressions only for $F^{1}_{P_3,1}$ and $F^{k\geqslant 2}_{P_3,1}$.

Then, $F^{1}_{P_3,1}$ is the sum of function $\mathcal{A}_{P_3}$ in (5.15), with $r=1$, over all possible values of $\{i_1,i_2,i_3\}$. Here, $\{i_1,i_2,i_3\}$ can take any distinct three values in the range from the start to the end of $\mathbf{R}_1$, i.e., from 0 to $(m+1)\gamma-1$ (see Fig. 5.6).

Moreover, $F^{k\geqslant 2}_{P_3,1}$ is the sum of function $\mathcal{B}_{P_3}$ in (5.16), with $r=1$ and $e=k$, over all possible values of $\{i_1,i_2,i_3\}$. Here, $\{i_1,i_2,i_3\}$ can take any distinct three values in the range from the start of $\mathbf{R}_k$ to the end of $\mathbf{R}_1$, i.e., from $(k-1)\gamma$ to $(m+1)\gamma-1$ (see also Fig. 5.6). $\square$

5.8.4 Proofs of Pattern P_4

Proof of Lemma 19

Proof. In Case 4.1, the number we are after is the number of ways to choose 4 overlaps out of $t_{\{i_1,i_2\}}$, which is given by (5.19). In Case 4.2, the number we are after is the number of ways to choose 3 overlaps out of $t_{\{i_1,i_2\}}$ and 1 overlap out of $t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma\}}$, which is given by (5.20). In Case 4.3, the number we are after is the number of ways to choose 2 overlaps out of $t_{\{i_1,i_2\}}$ and 2 overlaps out of $t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma\}}$, which is given by (5.21). In

Case 4.4, the number we are after is the number of ways to choose 2 overlaps out of $t_{\{i_1,i_2\}}$, 1 overlap out of $t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma\}}$, and 1 overlap out of $t_{\{i_1+(r-s)\gamma,i_2+(r-s)\gamma\}}$, which is given by (5.22). In Case 4.5, the number we are after is the number of ways to choose 1 overlap out of $t_{\{i_1,i_2\}}$, 1 overlap out of $t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma\}}$, 1 overlap out of $t_{\{i_1+(r-s)\gamma,i_2+(r-s)\gamma\}}$, and 1 overlap out of $t_{\{i_1+(r-u)\gamma,i_2+(r-u)\gamma\}}$, which is given by (5.23). □

Proof of Theorem 11

Proof. To compute F_{P_4}, we use the formula in [37, Theorem 1], with $\chi = m+1$. Since the overlaps of P_4 can exist in up to 4 replicas, we need to find expressions only for $F^1_{P_4,1}$, $F^2_{P_4,1}$, $F^3_{P_4,1}$, and $F^{k\geqslant 4}_{P_4,1}$.

Then, $F^1_{P_4,1}$ is the sum of function $\mathcal{A}_{P_4}$ in (5.19), with $r=1$, over all possible values of $\{i_1,i_2\}$. Here, $\{i_1,i_2\}$ can take any distinct two values in the range from 0 to $(m+1)\gamma-1$ (see Fig. 5.7).

Regarding $F^2_{P_4,1}$, we need to account for Case 4.2 and Case 4.3. For Case 4.2, we need to distinguish between two situations; when $r<e$ (i.e., replica $\mathbf{R}_r$, which has three overlaps, comes before replica $\mathbf{R}_e$), and when $r>e$ (i.e., replica $\mathbf{R}_r$ comes after replica $\mathbf{R}_e$). This distinction gives the two summations of function $\mathcal{B}_{P_4}$ in $F^2_{P_4,1}$. For the first summation, $\mathcal{B}_{P_4}$ in (5.20) has $r=1$ and $e=2$. Thus, $\{i_1,i_2\}$ can take any distinct two values in the range from γ to $(m+1)\gamma-1$. For the second summation, $\mathcal{B}_{P_4}$ in (5.20) has $r=2$ and $e=1$. Thus, $\{i_1,i_2\}$ can take any distinct two values in the range from 0 to $m\gamma-1$. The above distinction is not needed for Case 4.3 since the two replicas have the same number of degree-2 overlaps. For the third summation, $\mathcal{C}_{P_4}$ in (5.21) has $r=1$ and $e=2$. Thus, $\{i_1,i_2\}$ can take any distinct two values in the range from γ to $(m+1)\gamma-1$ (see Fig. 5.7 for more illustration).

As for $F^3_{P_4,1}$, the overlaps can be in 2 replicas (the first three summations in $F^3_{P_4,1}$) or 3 replicas (the following three summations in $F^3_{P_4,1}$). The first three summations are derived in a way similar to what we did for $F^2_{P_4,1}$, with a change in the summation indices; $\mathbf{R}_2$ is replaced by $\mathbf{R}_3$ here. The following three summations are related to Case 4.4. For Case 4.4, we need to distinguish between three situations; when $r < e < s$ (i.e., replica $\mathbf{R}_r$, which has two overlaps, comes before replicas $\mathbf{R}_e$ and $\mathbf{R}_s$ as in Fig. 5.7), when $e < r < s$ (i.e., replica $\mathbf{R}_r$ comes between replicas $\mathbf{R}_e$ and $\mathbf{R}_s$), and when $e < s < r$ (i.e., replica $\mathbf{R}_r$ comes after replicas $\mathbf{R}_e$ and $\mathbf{R}_s$). This distinction gives the three summations of function $\mathcal{D}_{P_4}$ in $F^3_{P_4,1}$. For the fourth summation in $F^3_{P_4,1}$, $\mathcal{D}_{P_4}$ in (5.22) has $r = 1$, $e = 2$, and $s = 3$. Thus, $\{i_1, i_2\}$ can take any distinct two values in the range from the start of $\mathbf{R}_3$ to the end of $\mathbf{R}_1$, i.e., from 2γ to $(m+1)\gamma - 1$. For the fifth summation, $\mathcal{D}_{P_4}$ in (5.22) has $r = 2$, $e = 1$, and $s = 3$. Thus, $\{i_1, i_2\}$ can take any distinct two values in the range from the start of $\mathbf{R}_3$ to the end of $\mathbf{R}_1$ ($\mathbf{R}_r$ now is $\mathbf{R}_2$), i.e., from γ to $m\gamma - 1$. For the sixth summation, $\mathcal{D}_{P_4}$ in (5.22) has $r = 3$, $e = 1$, and $s = 2$. Thus, $\{i_1, i_2\}$ can take any distinct two values in the range from the start of $\mathbf{R}_3$ (which is $\mathbf{R}_r$ now) to the end of $\mathbf{R}_1$, i.e., from 0 to $(m-1)\gamma - 1$.

Regarding $F^{k\geqslant 4}_{P_4,1}$, the overlaps can be in 2 replicas (the first three summations in $F^{k\geqslant 4}_{P_4,1}$), 3 replicas (the following three summations in $F^{k\geqslant 4}_{P_4,1}$), or 4 replicas (the seventh summation in $F^{k\geqslant 4}_{P_4,1}$). The first three summations are derived in a way similar to what we did for $F^2_{P_4,1}$, with a change in the summation indices; $\mathbf{R}_2$ is replaced by $\mathbf{R}_k$. The following three summations are derived in a way similar to what we did for $\mathcal{D}_{P_4}$ in $F^3_{P_4,1}$, with a change in the summation indices; $\mathbf{R}_2$ and $\mathbf{R}_3$ are replaced by $\mathbf{R}_h$ and $\mathbf{R}_k$, respectively, which also requires changing these three summations of $\mathcal{D}_{P_4}$ to be double summations. For the seventh (triple) summation, $\mathcal{E}_{P_4}$ in (5.23) has $r = 1$, $e = h$, $s = w$, and $u = k$. Thus, $\{i_1, i_2\}$ can take any distinct two values in the range from the start of $\mathbf{R}_k$ to the end of $\mathbf{R}_1$, i.e., from $(k-1)\gamma$ to $(m+1)\gamma - 1$ (see Fig. 5.7). The outer two summations are over all possible values of h and w, and we have $1 < h < k - 1$ and $h < w < k$. □

5.8.5 Proofs of Pattern P_5

Proof of Lemma 20

Proof. In Case 5.1, the number we are after is the number of ways to choose 2 overlaps out of $t_{\{i_1,i_2,i_3,i_4\}}$, which is given by (5.26). In Case 5.2, the number we are after is the number of ways to choose 1 overlap out of $t_{\{i_1,i_2,i_3,i_4\}}$ and 1 overlap out of $t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma,i_3+(r-e)\gamma,i_4+(r-e)\gamma\}}$, which is given in (5.27). □

Proof of Theorem 12

Proof. To compute F_{P_5}, we use the formula in [37, Theorem 1], with $\chi = m+1$. Since the overlaps of P_5 can exist in up to 2 replicas, we need to find expressions only for $F^{1}_{P_5,1}$ and $F^{k\geqslant 2}_{P_5,1}$.

Then, $F^{1}_{P_5,1}$ is the sum of function $\mathcal{A}_{P_5}$ in (5.26), with $r = 1$, over all possible values of $\{i_1, i_2, i_3, i_4\}$. Here, $\{i_1, i_2, i_3, i_4\}$ can take any distinct four values in the range from the start to the end of $\mathbf{R}_1$, i.e., from 0 to $(m+1)\gamma - 1$ (see Fig. 5.8).

Moreover, $F^{k\geqslant 2}_{P_5,1}$ is the sum of function $\mathcal{B}_{P_5}$ in (5.27), with $r = 1$ and $e = k$, over all possible values of $\{i_1, i_2, i_3, i_4\}$. Here, $\{i_1, i_2, i_3, i_4\}$ can take any distinct four values in the range from the start of $\mathbf{R}_k$ to the end of $\mathbf{R}_1$, i.e., from $(k-1)\gamma$ to $(m+1)\gamma - 1$ (see also Fig. 5.8). □

5.8.6 Proofs of Pattern P_6

Proof of Lemma 21

Proof. In Case 6.1, the number we are after is the number of ways to choose 1 overlap from each family in $\mathbf{R}_r$ (there exist three different families for P_6). We choose the $c_1 - c_2 - c_3$ degree-3 overlap first. Then, in order to avoid over-counting, it is required to distinguish between the two situations when the $c_1 - c_2$ degree-2 overlap is part of a $c_1 - c_2 - c_3$ degree-3 overlap, and when this is not the case. Taking this requirement into account yields the two added terms in (5.30). The same applies for Case 6.2, with the exception that here the degree-3 overlap is chosen from $t_{\{i_1+(r-e)\gamma,i_2+(r-e)\gamma,i_3+(r-e)\gamma\}}$ overlaps, resulting in (5.31). Following the same logic of Case 6.1 for Case 6.3, with the exception that the $c_1 - c_3$ overlap is chosen from $t_{\{i_1+(r-e)\gamma,i_3+(r-e)\gamma\}}$ overlaps, gives (5.32). In Case 6.4, the number we are after is the number of ways to choose 1 overlap out of $t_{\{i_1,i_2\}}$, 1 overlap out of $t_{\{i_1+(r-e)\gamma, i_3+(r-e)\gamma\}}$, and 1 overlap out of $t_{\{i_1+(r-s)\gamma, i_2+(r-s)\gamma, i_3+(r-s)\gamma\}}$, which is given by (5.33). $\square$

Proof of Theorem 13

Proof. To compute F_{P_6}, we use the formula in [37, Theorem 1], with $\chi = m + 1$. Since the overlaps of P_6 can exist in up to 3 replicas, we need to find expressions only for $F^1_{P_6,1}$, $F^2_{P_6,1}$, and $F^{k\geqslant 3}_{P_6,1}$.

Then, $F^1_{P_6,1}$ is the sum of function $\mathcal{A}_{P_6}$ in (5.30), with $r = 1$, over all possible values of i_1 and $\{i_2, i_3\}$. In Pattern P_6, CN c_1, which connects all three VNs, is different from the other two CNs. Moreover, in a group of three CNs that can form P_6, c_1 can be any one of these three CNs, which means we have three possible ways to form P_6 from these three CNs. These facts combined are the reason why i_1 of c_1 has to be separated from $\{i_2, i_3\}$, despite having the same range, in the expression of $F^1_{P_6,1}$ (this applies for other expressions too). Here, i_1 (resp., $\{i_2, i_3\}$) can take any value (resp., distinct two values) in the range from the start to the end of $\mathbf{R}_1$, i.e., from 0 to $(m + 1)\gamma - 1$ (see also Fig. 5.9).

Regarding $F^2_{P_6,1}$, we need to account for Case 6.2 and Case 6.3. For each case of the two, we need to distinguish between two situations; when $r < e$ and when $r > e$. This distinction gives the two summations of $\mathcal{B}_{P_6}$ and the two summations of $\mathcal{C}_{P_6}$ in $F^2_{P_6,1}$. In Case 6.2, each of the three CNs of P_6 connects overlaps in $\mathbf{R}_r$ and $\mathbf{R}_e$ (because the degree-3 overlap is moved to $\mathbf{R}_e$). For the first summation in $F^2_{P_6,1}$, $\mathcal{B}_{P_6}$ in (5.31) has $r = 1$ and $e = 2$. Thus, i_1 (resp., $\{i_2, i_3\}$) can take any value (resp., distinct two values) in the range from the start of $\mathbf{R}_2$ to the end of $\mathbf{R}_1$, i.e., from γ to $(m + 1)\gamma - 1$. For the second summation, $\mathcal{B}_{P_6}$ in (5.31) has $r = 2$ and $e = 1$. Thus, i_1 (resp., $\{i_2, i_3\}$) can take any value (resp., distinct two values) in the range from 0 to $m\gamma - 1$. In Case 6.3, and as shown in Fig. 5.9, c_1 and c_3 each connects overlaps in $\mathbf{R}_r$ and $\mathbf{R}_e$, while c_2 connects overlaps in $\mathbf{R}_r$ only (because the $c_1 - c_3$ overlap is moved to $\mathbf{R}_e$ here). For the third summation in $F^2_{P_6,1}$, $\mathcal{C}_{P_6}$ in (5.32) has $r = 1$ and $e = 2$. Thus, i_1 (resp., i_2 and i_3) can take any value in the range from the start of $\mathbf{R}_2$ (resp., $\mathbf{R}_1$

and $\mathbf{R}_2$) to the end of $\mathbf{R}_1$, i.e., from γ (resp., 0 and γ) to $(m+1)\gamma - 1$. For the fourth summation, $\mathcal{C}_{P_6}$ in (5.32) has $r = 2$ and $e = 1$ (see Fig. 5.9). Thus, i_1 (resp., i_2 and i_3) can take any value in the range from the start of $\mathbf{R}_2$ to the end of $\mathbf{R}_1$ (resp., $\mathbf{R}_2$ and $\mathbf{R}_1$), i.e., from 0 to $m\gamma - 1$ (resp., $(m+1)\gamma - 1$ and $m\gamma - 1$). Note that the ranges of i_2 and i_3 are different in Case 6.3, unlike Case 6.2, which is the reason why i_2 and i_3 are not in a set in the summations of $\mathcal{C}_{P_6}$.

As for $F_{P_6,1}^{k\geqslant 3}$, the overlaps can be in 2 replicas (the first four summations in $F_{P_6,1}^{k\geqslant 3}$) or 3 replicas (the following three summations in $F_{P_6,1}^{k\geqslant 3}$). The first four summations are derived in a way similar to what we did for $F_{P_6,1}^{2}$, with a change in the summation indices; $\mathbf{R}_2$ is replaced by $\mathbf{R}_k$ here. The following three summations are associated with Case 6.4. In Case 6.4, c_1 connects overlaps in $\mathbf{R}_r$, $\mathbf{R}_e$, and $\mathbf{R}_s$. On the other hand, c_2 (resp., c_3) connects overlaps in $\mathbf{R}_r$ (resp., $\mathbf{R}_e$) and $\mathbf{R}_s$. For the fifth (double) summation, $\mathcal{D}_{P_6}$ in (5.33) has $r = 1$, $e = h$, and $s = k$ (see Fig. 5.9). Thus, i_1 (resp., i_2 and i_3) can take any value in the range from the start of $\mathbf{R}_k$ to the end of $\mathbf{R}_1$ (resp., $\mathbf{R}_1$ and $\mathbf{R}_h$), i.e., from $(k-1)\gamma$ to $(m+1)\gamma - 1$ (resp., $(m+1)\gamma - 1$ and $(m+h)\gamma - 1$). For the sixth (double) summation, $\mathcal{D}_{P_6}$ in (5.33) has $r = 1$, $e = k$, and $s = h$. Thus, i_1 (resp., i_2 and i_3) can take any value in the range from the start of $\mathbf{R}_k$ (resp., $\mathbf{R}_h$ and $\mathbf{R}_k$) to the end of $\mathbf{R}_1$ (resp., $\mathbf{R}_1$ and $\mathbf{R}_h$), i.e., from $(k-1)\gamma$ (resp., $(h-1)\gamma$ and $(k-1)\gamma$) to $(m+1)\gamma - 1$ (resp., $(m+1)\gamma - 1$ and $(m+h)\gamma - 1$). For the seventh (double) summation, $\mathcal{D}_{P_6}$ in (5.33) has $r = h$, $e = k$, and $s = 1$. Thus, i_1 (resp., i_2 and i_3) can take any value in the range from the start of $\mathbf{R}_k$ (resp., $\mathbf{R}_h$ and $\mathbf{R}_k$) to the end of $\mathbf{R}_1$, i.e., from $(k-h)\gamma$ (resp., 0 and $(k-h)\gamma$) to $(m-h+2)\gamma - 1$. The outer summation is over all possible values of h, and we have $1 < h < k$. □

5.8.7 Proofs of Pattern P_7

Proof of Lemma 22

Proof. In Case 7.1, the number we are after is the number of ways to choose 2 overlaps from each family in $\mathbf{R}_r$ (the pattern has two $c_1 - c_2$ overlaps and two $c_1 - c_3$ overlaps). In order to avoid over-counting, it is required to distinguish between the three situations when the two $c_1 - c_2$ overlaps are each part of a $c_1 - c_2 - c_3$ overlap, when only one $c_1 - c_2$ overlap is part of a $c_1 - c_2 - c_3$ overlap, and when neither of them is. Taking this requirement into account yields the three added terms in (5.36). The same applies for Case 7.2, with the exception that here, one $c_1 - c_3$ overlap is chosen from $t_{\{i_1+(r-e)\gamma, i_3+(r-e)\gamma\}}$ overlaps. In Case 7.3, there is no need to make this distinction since both $c_1 - c_3$ overlaps are chosen from $t_{\{i_1+(r-e)\gamma, i_3+(r-e)\gamma\}}$ overlaps (they are in $\mathbf{R}_e$), and the result is in (5.38). In Case 7.4, the distinction is applied separately on the $c_1 - c_2$ overlap in $\mathbf{R}_r$ and the $c_1 - c_2$ overlap in $\mathbf{R}_e$ to give (5.39). Case 7.5 is similar to Case 7.3, with the exception that one of the two $c_1 - c_3$ overlaps is chosen from $t_{\{i_1+(r-s)\gamma, i_3+(r-s)\gamma\}}$ overlaps since it is now in $\mathbf{R}_s$. Case 7.6 is similar to Case 7.4, with the exception that one $c_1 - c_3$ overlap is chosen from $t_{\{i_1+(r-s)\gamma, i_3+(r-s)\gamma\}}$ overlaps since it is now in $\mathbf{R}_s$ (was the $c_1 - c_3$ overlap in $\mathbf{R}_e$ in Case 7.4). Consequently, the above distinction is only applied to the $c_1 - c_2$ overlap in $\mathbf{R}_r$, which results in (5.41). In Case 7.7, the number we are after is the number of ways to choose 1 overlap out of $t_{\{i_1,i_2\}}$, 1 overlap out of $t_{\{i_1+(r-e)\gamma, i_2+(r-e)\gamma\}}$, 1 overlap out of $t_{\{i_1+(r-s)\gamma, i_3+(r-s)\gamma\}}$, and 1 overlap out of $t_{\{i_1+(r-u)\gamma, i_3+(r-u)\gamma\}}$, which is given by (5.42). □

Proof of Theorem 14

Proof. To compute F_{P_7}, we use the formula in [37, Theorem 1], with $\chi = m + 1$. Since the overlaps of P_7 can exist in up to 4 replicas, we need to find expressions only for $F^1_{P_7,1}$, $F^2_{P_7,1}$, $F^3_{P_7,1}$, and $F^{k\geqslant 4}_{P_7,1}$.

Then, $F^1_{P_7,1}$ is the sum of function $\mathcal{A}_{P_7}$ in (5.36), with $r = 1$, over all possible values of i_1 and $\{i_1, i_2\}$. In Pattern P_7, CN c_1, which connects all four VNs, is different from the other two CNs. Moreover, in a group of three CNs that can form P_7, c_1 can be any one of these three CNs, which means we have three possible ways to form P_7 from these three CNs. These facts combined are the reason why i_1 of c_1 has to be separated from $\{i_2, i_3\}$, despite having the same range, in the expression of $F^1_{P_7,1}$ (this applies for other expressions too).

Here, i_1 (resp., $\{i_2, i_3\}$) can take any value (resp., distinct two values) in the range from the start to the end of $\mathbf{R}_1$, i.e., from 0 to $(m+1)\gamma - 1$ (see also Fig. 5.10).

Regarding $F^2_{P_7,1}$, we need to account for Case 7.2, Case 7.3, and Case 7.4. For Case 7.2, we need to distinguish between two situations; when $r < e$ and when $r > e$, which gives the two summations of $\mathcal{B}_{P_7}$ in $F^2_{P_7,1}$. In Case 7.2, and as shown in Fig. 5.10, c_1 and c_3 each connects overlaps in $\mathbf{R}_r$ and $\mathbf{R}_e$, while c_2 connects overlaps in $\mathbf{R}_r$ only. For the first summation in $F^2_{P_7,1}$, $\mathcal{B}_{P_7}$ in (5.37) has $r = 1$ and $e = 2$. Thus, i_1 (resp., i_2 and i_3) can take any value in the range from the start of $\mathbf{R}_2$ (resp., $\mathbf{R}_1$ and $\mathbf{R}_2$) to the end of $\mathbf{R}_1$, i.e., from γ (resp., 0 and γ) to $(m+1)\gamma - 1$. For the second summation, $\mathcal{B}_{P_7}$ in (5.37) has $r = 2$ and $e = 1$. Thus, i_1 (resp., i_2 and i_3) can take any value in the range from the start of $\mathbf{R}_2$ to the end of $\mathbf{R}_1$ (resp., $\mathbf{R}_2$ and $\mathbf{R}_1$), i.e., from 0 to $m\gamma - 1$ (resp., $(m+1)\gamma - 1$ and $m\gamma - 1$). Note that the ranges of i_2 and i_3 are different in Case 7.2. The above distinction is not needed for neither Case 7.3 nor Case 7.4 since the two replicas have the same number of degree-2 overlaps with similar connectivity. In Case 7.3, c_1 connects overlaps in $\mathbf{R}_r$ and $\mathbf{R}_e$, while c_2 (resp., c_3) connects overlaps in $\mathbf{R}_r$ (resp., $\mathbf{R}_e$) only. For the third summation in $F^2_{P_7,1}$, $\mathcal{C}_{P_7}$ in (5.38) has $r = 1$ and $e = 2$. Thus, i_1 (resp., i_2 and i_3) can take any value in the range from the start of $\mathbf{R}_2$ (resp., $\mathbf{R}_1$ and $\mathbf{R}_2$) to the end of $\mathbf{R}_1$ (resp., $\mathbf{R}_1$ and $\mathbf{R}_2$), i.e., from γ (resp., 0 and γ) to $(m+1)\gamma - 1$ (resp., $(m+1)\gamma - 1$ and $(m+2)\gamma - 1$). Note that the ranges of i_2 and i_3 are also different in Case 7.3. In Case 7.4, all the CNs connect overlaps in $\mathbf{R}_r$ and $\mathbf{R}_e$. For the fourth summation in $F^2_{P_7,1}$, $\mathcal{D}_{P_7}$ in (5.39) has $r = 1$ and $e = 2$. Thus, i_1 (resp., $\{i_2, i_3\}$) can take any value (resp., distinct two values) in the range from the start of $\mathbf{R}_2$ to the end of $\mathbf{R}_1$, i.e., from γ to $(m+1)\gamma - 1$. Note that the ranges of i_2 and i_3 are the same in Case 7.4 (similar to Case 7.1).

As for $F^3_{P_7,1}$, the overlaps can be in 2 replicas (the first four summations in $F^3_{P_7,1}$) or 3 replicas (the following six summations in $F^3_{P_7,1}$). The first four summations are derived in a way similar to what we did for $F^2_{P_7,1}$, with a change in the summation indices; $\mathbf{R}_2$ is replaced by $\mathbf{R}_3$ here. Then, we need to account for Case 7.5 (fifth to seventh summations) and Case 7.6 (eighth to tenth summations). In Case 7.5, c_1 connects overlaps in $\mathbf{R}_r$, $\mathbf{R}_e$, and $\mathbf{R}_s$. On the other hand, c_2 (resp., c_3) connects overlaps in $\mathbf{R}_r$ only (resp., $\mathbf{R}_e$ and $\mathbf{R}_s$). For the fifth summation, $\mathcal{E}_{P_7}$ in (5.40) has $r = 1$, $e = 2$, and $s = 3$ (see Fig. 5.10). Thus, i_1 (resp., i_2 and i_3) can take any value in the range from the start of $\mathbf{R}_3$ (resp., $\mathbf{R}_1$ and $\mathbf{R}_3$) to the end of $\mathbf{R}_1$ (resp., $\mathbf{R}_1$ and $\mathbf{R}_2$), i.e., from 2γ (resp., 0 and 2γ) to $(m+1)\gamma - 1$ (resp., $(m+1)\gamma - 1$ and $(m+2)\gamma - 1$). For the sixth summation, $\mathcal{E}_{P_7}$ in (5.40) has $r = 2$, $e = 1$, and $s = 3$. Thus, i_1 (resp., i_2 and i_3) can take any value in the range from the start of $\mathbf{R}_3$ (resp., $\mathbf{R}_2$ and $\mathbf{R}_3$) to the end of $\mathbf{R}_1$ (resp., $\mathbf{R}_2$ and $\mathbf{R}_1$), i.e., from γ (resp., 0 and γ) to $m\gamma - 1$ (resp., $(m+1)\gamma - 1$ and $m\gamma - 1$). For the seventh summation, $\mathcal{E}_{P_7}$ in (5.40) has $r = 3$, $e = 1$, and $s = 2$. Thus, i_1 (resp., i_2 and i_3) can take any value in the range from the start of $\mathbf{R}_3$ (resp., $\mathbf{R}_3$ and $\mathbf{R}_2$) to the end of $\mathbf{R}_1$ (resp., $\mathbf{R}_3$ and $\mathbf{R}_1$), i.e., from 0 (resp., 0 and $-\gamma$) to $(m-1)\gamma - 1$ (resp., $(m+1)\gamma - 1$ and $(m-1)\gamma - 1$). In Case 7.6, c_1 connects overlaps in $\mathbf{R}_r$, $\mathbf{R}_e$, and $\mathbf{R}_s$. On the other hand, c_2 (resp., c_3) connects overlaps in $\mathbf{R}_r$ and $\mathbf{R}_e$ (resp., $\mathbf{R}_s$). For the eighth summation, $\mathcal{G}_{P_7}$ in (5.41) has $r = 1$, $e = 2$, and $s = 3$. Thus, i_1 (resp., i_2 and i_3) can take any value in the range from the start of $\mathbf{R}_3$ (resp., $\mathbf{R}_2$ and $\mathbf{R}_3$) to the end of $\mathbf{R}_1$, i.e., from 2γ (resp., γ and 2γ) to $(m+1)\gamma - 1$. For the ninth summation, $\mathcal{G}_{P_7}$ in (5.41) has $r = 2$, $e = 1$, and $s = 3$. Thus, i_1 (resp., i_2 and i_3) can take any value in the range from the start of $\mathbf{R}_3$ (resp., $\mathbf{R}_2$ and $\mathbf{R}_3$) to the end of $\mathbf{R}_1$ (resp., $\mathbf{R}_1$ and $\mathbf{R}_2$), i.e., from γ (resp., 0 and γ) to $m\gamma - 1$ (resp., $m\gamma - 1$ and $(m+1)\gamma - 1$). For the tenth summation, $\mathcal{G}_{P_7}$ in (5.41) has $r = 3$, $e = 1$, and $s = 2$. Thus, i_1 (resp., i_2 and i_3) can take any value in the range from the start of $\mathbf{R}_3$ to the end of $\mathbf{R}_1$ (resp., $\mathbf{R}_1$ and $\mathbf{R}_2$), i.e., from 0 to $(m-1)\gamma - 1$ (resp., $(m-1)\gamma - 1$ and $m\gamma - 1$).

Regarding $F_{P_7,1}^{k\geqslant 4}$, the overlaps can be in 2 replicas (the first four summations in $F_{P_7,1}^{k\geqslant 4}$), 3 replicas (the following six summations in $F_{P_7,1}^{k\geqslant 4}$), or 4 replicas (the last three summations in $F_{P_7,1}^{k\geqslant 4}$). The first four summations are derived in a way similar to what we did for $F_{P_7,1}^{2}$, with a change in the summation indices; $\mathbf{R}_2$ is replaced by $\mathbf{R}_k$. The following six summations are derived in a way similar to what we did for $F_{P_7,1}^{3}$, with a change in the summation indices;

$\mathbf{R}_2$ and $\mathbf{R}_3$ are replaced by $\mathbf{R}_h$ and $\mathbf{R}_k$, respectively, which also requires changing these six summations of $\mathcal{E}_{P_7}$ and $\mathcal{G}_{P_7}$ to be double summations. The following three summations are associated with Case 7.7. In Case 7.7, c_1 connects overlaps in $\mathbf{R}_r$, $\mathbf{R}_e$, $\mathbf{R}_s$, and $\mathbf{R}_u$. On the other hand, c_2 (resp., c_3) connects overlaps in $\mathbf{R}_r$ and $\mathbf{R}_e$ (resp., $\mathbf{R}_s$ and $\mathbf{R}_u$). See Fig. 5.10 for more illustration. There are three situations to distinguish between; the two $c_1 - c_2$ overlaps are in the first and second replicas, in the first and third replicas, and in the first and last replicas. The ordering of replicas here is with respect to the four replicas in which the overlaps of P_7 exist. For the eleventh (triple) summation, $\mathcal{I}_{P_7}$ in (5.42) has $r = 1$, $e = h$, $s = w$, and $u = k$. Thus, i_1 (resp., i_2 and i_3) can take any value in the range from the start of $\mathbf{R}_k$ (resp., $\mathbf{R}_h$ and $\mathbf{R}_k$) to the end of $\mathbf{R}_1$ (resp., $\mathbf{R}_1$ and $\mathbf{R}_w$), i.e., from $(k-1)\gamma$ (resp., $(h-1)\gamma$ and $(k-1)\gamma$) to $(m+1)\gamma - 1$ (resp., $(m+1)\gamma - 1$ and $(m+w)\gamma - 1$). For the twelfth (triple) summation, $\mathcal{I}_{P_7}$ in (5.42) has $r = 1$, $e = w$, $s = h$, and $u = k$. Thus, i_1 (resp., i_2 and i_3) can take any value in the range from the start of $\mathbf{R}_k$ (resp., $\mathbf{R}_w$ and $\mathbf{R}_k$) to the end of $\mathbf{R}_1$ (resp., $\mathbf{R}_1$ and $\mathbf{R}_h$), i.e., from $(k-1)\gamma$ (resp., $(w-1)\gamma$ and $(k-1)\gamma$) to $(m+1)\gamma - 1$ (resp., $(m+1)\gamma - 1$ and $(m+h)\gamma - 1$). For the thirteenth (triple) summation, $\mathcal{I}_{P_7}$ in (5.42) has $r = 1$, $e = k$, $s = h$, and $u = w$. Thus, i_1 (resp., i_2 and i_3) can take any value in the range from the start of $\mathbf{R}_k$ (resp., $\mathbf{R}_k$ and $\mathbf{R}_w$) to the end of $\mathbf{R}_1$ (resp., $\mathbf{R}_1$ and $\mathbf{R}_h$), i.e., from $(k-1)\gamma$ (resp., $(k-1)\gamma$ and $(w-1)\gamma$) to $(m+1)\gamma - 1$ (resp., $(m+1)\gamma - 1$ and $(m+h)\gamma - 1$). The outer two summations are over all possible values of h and w, and we have $1 < h < k-1$ and $h < w < k$ (similar to Pattern P_4).

Note that c_2 and c_3 are not adjacent (no path of only one VN connects them) in P_7, which means it is possible to have $\overline{i_2} = \overline{i_3}$, but not $i_2 = i_3$, for that pattern. □

5.8.8 Proofs of Pattern P_8

Proof of Lemma 23

Proof. In Case 8.1, the number we are after is the number of ways to choose 1 overlap from each family in $\mathbf{R}_r$ (there exist three different families for P_8). We choose the $c_1 - c_2 - c_3 - c_4$

degree-4 overlap first. Then, in order to avoid over-counting, it is required to distinguish between the two situations when the $c_1 - c_2$ degree-2 overlap is part of a $c_1 - c_2 - c_3 - c_4$ degree-4 overlap, and when this is not the case. Taking this requirement into account yields the two added terms in (5.45). The same applies for Case 8.2, with the exception that here the degree-4 overlap is chosen from $t_{\{i_1+(r-e)\gamma, i_2+(r-e)\gamma, i_3+(r-e)\gamma\, i_4+(r-e)\gamma\}}$ overlaps, resulting in (5.46). Following the same logic of Case 8.1 for Case 8.3, with the exception that the $c_3 - c_4$ overlap is chosen from $t_{\{i_3+(r-e)\gamma, i_4+(r-e)\gamma\}}$ overlaps, gives (5.47). In Case 8.4, the number we are after is the number of ways to choose 1 overlap out of $t_{\{i_1,i_2\}}$, 1 overlap out of $t_{\{i_3+(r-e)\gamma, i_4+(r-e)\gamma\}}$, and 1 overlap out of $t_{\{i_1+(r-s)\gamma, i_2+(r-s)\gamma, i_3+(r-s)\gamma, i_4+(r-s)\gamma\}}$, which is given by (5.48). □

Proof of Theorem 15

Proof. To compute F_{P_8}, we use the formula in [37, Theorem 1], with $\chi = 2m + 1$. Since the overlaps of P_8 can exist in up to 3 replicas, we need to find expressions only for $F^1_{P_8,1}$, $F^2_{P_8,1}$, and $F^{k\geqslant 3}_{P_8,1}$.

Then, $F^1_{P_8,1}$ is the sum of function $\mathcal{A}_{P_8}$ in (5.45), with $r = 1$, over all possible values of $\{i_1, i_2\}$ and $\{i_3, i_4\}$. In Pattern P_8, CNs c_1 and c_2 are directly connected twice, and CNs c_3 and c_4 are directly connected twice, which creates two separate groups of CNs. Moreover, in a group of four CNs that can form P_8, c_1 and c_2 can be any two of these four CNs. These facts combined are the reason why the set $\{i_1, i_2\}$ has to be separated from the set $\{i_3, i_4\}$, despite having the same range, in the expression of $F^1_{P_8,1}$ (this applies for other expressions too). We have $\binom{4}{2} = 6$ possible ways to choose $\{i_1, i_2\}$, i.e., to choose c_1 and c_2 out of the four CNs. However, it does not matter for the count of $\mathcal{A}_{P_8}$ whether the set $\{i_1, i_2\}$ or the set $\{i_3, i_4\}$ is chosen first. Thus, we only have three possible ways to form P_8 from these four CNs, and the remaining three ways are repetitive. This fact is the reason why we multiply by $\frac{1}{2}$ in the expression $F^1_{P_8,1}$. Here, $\{i_1, i_2\}$ (resp., $\{i_3, i_4\}$) can take any distinct two values in the range from the start to the end of $\mathbf{R}_1$, i.e., from 0 to $(m+1)\gamma - 1$ (see also Fig. 5.11).

Regarding $F^2_{P_8,1}$, we need to account for Case 8.2 and Case 8.3. For each case of the two, we need to distinguish between two situations; when $r < e$ and when $r > e$. This distinction gives the two summations of $\mathcal{B}_{P_8}$ and the two summations of $\mathcal{C}_{P_8}$ in $F^2_{P_8,1}$. In Case 8.2, the multiplication by $\frac{1}{2}$ for the counts of $\mathcal{B}_{P_8}$ is also to account for repetitions (as with $\mathcal{A}_{P_8}$). Moreover, in Case 8.2, each of the four CNs of P_8 connects overlaps in $\mathbf{R}_r$ and $\mathbf{R}_e$ (because the degree-4 overlap is moved to $\mathbf{R}_e$), as shown in Fig. 5.11. For the first summation in $F^2_{P_8,1}$, $\mathcal{B}_{P_8}$ in (5.46) has $r = 1$ and $e = 2$ (see also Fig. 5.11). Thus, $\{i_1, i_2\}$ (resp., $\{i_3, i_4\}$) can take any distinct two values in the range from the start of $\mathbf{R}_2$ to the end of $\mathbf{R}_1$, i.e., from γ to $(m+1)\gamma - 1$. For the second summation, $\mathcal{B}_{P_8}$ in (5.46) has $r = 2$ and $e = 1$. Thus, $\{i_1, i_2\}$ (resp., $\{i_3, i_4\}$) can take any distinct two values in the range from 0 to $m\gamma - 1$. In Case 8.3, and contrarily to Case 8.2, it does matter for the count of $\mathcal{C}_{P_8}$ whether the set $\{i_1, i_2\}$ or the set $\{i_3, i_4\}$ is chosen first because the degree-2 overlaps, $c_1 - c_2$ and $c_3 - c_4$, are in two different replicas. Consequently, the multiplication by $\frac{1}{2}$ is not needed in this case. Moreover, in Case 8.3, c_1 and c_2 each connects overlaps in $\mathbf{R}_r$ only, while c_3 and c_4 each connects overlaps in $\mathbf{R}_r$ and $\mathbf{R}_e$ (because the $c_3 - c_4$ overlap is moved to $\mathbf{R}_e$ here). For the third summation in $F^2_{P_8,1}$, $\mathcal{C}_{P_8}$ in (5.47) has $r = 1$ and $e = 2$. Thus, $\{i_1, i_2\}$ (resp., $\{i_3, i_4\}$) can take any distinct two values in the range from the start of $\mathbf{R}_1$ (resp., $\mathbf{R}_2$) to the end of $\mathbf{R}_1$, i.e., from 0 (resp., γ) to $(m+1)\gamma - 1$. For the fourth summation, $\mathcal{C}_{P_8}$ in (5.47) has $r = 2$ and $e = 1$. Thus, $\{i_1, i_2\}$ (resp., $\{i_3, i_4\}$) can take any distinct two values in the range from the start of $\mathbf{R}_2$ to the end of $\mathbf{R}_2$ (resp., $\mathbf{R}_1$), i.e., from 0 to $(m+1)\gamma - 1$ (resp., $m\gamma - 1$).

As for $F^{k\geqslant 3}_{P_8,1}$, the overlaps can be in 2 replicas (the first four summations in $F^{k\geqslant 3}_{P_8,1}$) or 3 replicas (the following three summations in $F^{k\geqslant 3}_{P_8,1}$). The first four summations are derived in a way similar to what we did for $F^2_{P_8,1}$, with a change in the summation indices; $\mathbf{R}_2$ is replaced by $\mathbf{R}_k$ here. The following three summations are associated with Case 8.4. In Case 8.4, c_1 and c_2 each connects overlaps in $\mathbf{R}_r$ and $\mathbf{R}_s$. On the other hand, c_3 and c_4 each connects overlaps in $\mathbf{R}_e$ and $\mathbf{R}_s$. For the fifth (double) summation, $\mathcal{D}_{P_8}$ in (5.48) has $r = 1$, $e = h$, and $s = k$. Thus, $\{i_1, i_2\}$ (resp., $\{i_3, i_4\}$) can take any distinct two values in the range from the start of $\mathbf{R}_k$ to the end of $\mathbf{R}_1$ (resp., $\mathbf{R}_h$), i.e., from $(k-1)\gamma$ to $(m+1)\gamma-1$ (resp., $(m+h)\gamma-1$). For the sixth (double) summation, $\mathcal{D}_{P_6}$ in (5.48) has $r=1$, $e=k$, and $s=h$ (see Fig. 5.11). Thus, $\{i_1, i_2\}$ (resp., $\{i_3, i_4\}$) can take any distinct two values in the range from the start of $\mathbf{R}_h$ (resp., $\mathbf{R}_k$) to the end of $\mathbf{R}_1$ (resp., $\mathbf{R}_h$), i.e., from $(h-1)\gamma$ (resp., $(k-1)\gamma$) to $(m+1)\gamma-1$ (resp., $(m+h)\gamma-1$). For the seventh (double) summation, $\mathcal{D}_{P_8}$ in (5.48) has $r=h$, $e=k$, and $s=1$. Thus, $\{i_1, i_2\}$ (resp., $\{i_3, i_4\}$) can take any distinct two values in the range from the start of $\mathbf{R}_h$ (resp., $\mathbf{R}_k$) to the end of $\mathbf{R}_1$, i.e., from 0 (resp., $(k-h)\gamma$) to $(m-h+2)\gamma-1$. The outer summation is over all possible values of h, and we have $1 < h < k$. □

5.8.9 Proofs of Pattern P_9

Proof of Lemma 24

Proof. In Case 9.1, the number we are after is the number of ways to choose 1 overlap from each family in $\mathbf{R}_r$ (there exist four different families for P_9). In order to avoid over-counting, multiple distinctions need to be performed. For the degree-2 overlap c_1-c_2, it is required to distinguish between the four situations when that overlap is part of a $c_1-c_2-c_3-c_4$ degree-4 overlap, when that overlap is part of a $c_1-c_2-c_3$ degree-3 overlap that is not itself part of a $c_1-c_2-c_3-c_4$ degree-4 overlap, when that overlap is part of a $c_1-c_2-c_4$ degree-3 overlap that is not itself part of a $c_1-c_2-c_3-c_4$ degree-4 overlap, and when neither of these previous three situations holds. This particular distinction results in having four functions, $\mathcal{A}_{P_9,1}$, $\mathcal{A}_{P_9,2}$, $\mathcal{A}_{P_9,3}$, and $\mathcal{A}_{P_9,4}$. Next, only the degree-2 overlaps c_2-c_3, c_3-c_4, and c_1-c_4 need to be chosen. Consequently, for the degree-2 overlap c_2-c_3, it is required to distinguish between only three situations; when that overlap is part of a $c_1-c_2-c_3-c_4$ degree-4 overlap, when that overlap is part of a $c_2-c_3-c_4$ degree-3 overlap that is not itself part of a $c_1-c_2-c_3-c_4$ degree-4 overlap, and when neither of these previous two situations holds. As for the degree-2 overlap c_3-c_4, it is required to distinguish between only two situations; when that overlap is part of a $c_1-c_3-c_4$ degree-3 overlap, and when this is not the case. Addressing all these distinctions results in (5.51) and (5.52), with six added terms for each of the four functions constituting $\mathcal{A}_{P_9}$.

The same applies for Case 9.2, with the exception that here the degree-2 overlap $c_1 - c_4$ is chosen from $t_{\{i_1+(r-e)\gamma, i_4+(r-e)\gamma\}}$ overlaps, which divides the number of added terms in (5.52) by four to reach (5.53). In Case 9.3, the distinction is applied separately on the $c_1 - c_2$ overlap in $\mathbf{R}_r$ and the $c_3 - c_4$ overlap in $\mathbf{R}_e$ to give (5.54). The distinction here is between two situations; when the degree-2 overlap is part of a degree-3 overlap, and when this is not the case. In Case 9.4, the distinction is applied separately on the $c_1 - c_2$ overlap in $\mathbf{R}_r$ and the $c_2 - c_3$ overlap in $\mathbf{R}_e$ to give (5.55). The distinction here is between two situations; when the degree-2 overlap is part of a degree-4 overlap, and when this is not the case. Case 9.5 is similar to Case 6.2, with the exception that here there are two degree-2 overlaps outside $\mathbf{R}_r$, and they are distributed over $\mathbf{R}_e$ (for the $c_3 - c_4$ overlap) and $\mathbf{R}_s$ (for the $c_1 - c_4$ overlap). Case 9.6 is similar to Case 8.2, with the exception that here there are two degree-2 overlaps outside $\mathbf{R}_r$, and they are distributed over $\mathbf{R}_e$ (for the $c_2 - c_3$ overlap) and $\mathbf{R}_s$ (for the $c_1 - c_4$ overlap). In Case 9.7, the number we are after is the number of ways to choose 1 overlap out of $t_{\{i_1,i_2\}}$, 1 overlap out of $t_{\{i_2+(r-e)\gamma, i_3+(r-e)\gamma\}}$, 1 overlap out of $t_{\{i_3+(r-s)\gamma, i_4+(r-s)\gamma\}}$, and 1 overlap out of $t_{\{i_1+(r-u)\gamma, i_4+(r-u)\gamma\}}$, which is given by (5.58). $\square$

Proof of Theorem 16

Proof. To compute F_{P_9}, we use the formula in [37, Theorem 1], with $\chi = 2m+1$. Since the overlaps of P_9 can exist in up to 4 replicas, we need to find expressions only for $F^1_{P_9,1}$, $F^2_{P_9,1}$, $F^3_{P_9,1}$, and $F^{k\geqslant 4}_{P_9,1}$.

Then, $F^1_{P_9,1}$ is the sum of function $\mathcal{A}_{P_9}$ in (5.51), with $r = 1$, over all possible values of $\{i_1, i_3\}$ and $\{i_2, i_4\}$. In a group of four CNs, say c_{x_1}, c_{x_2}, c_{x_3}, and c_{x_4}, there exist 3 unique ways to form P_9, which is a cycle of length 8, among them. These 3 ways are: $c_{x_1} - c_{x_2} - c_{x_3} - c_{x_4}$, $c_{x_1} - c_{x_2} - c_{x_4} - c_{x_3}$, and $c_{x_1} - c_{x_3} - c_{x_2} - c_{x_4}$. In these ways, VNs are omitted for convenience, and the last CN in each way is connected to the first CN through a VN. These facts combined are the reason why we separate $\{i_1, i_3\}$ from $\{i_2, i_4\}$, despite having the same range, in the expression of $F^1_{P_9,1}$ (this applies for other expressions too). Since this separation gives $\binom{4}{2} = 6$ options, we multiply by $\frac{1}{2}$ in the expression of $F^1_{P_9,1}$ to account for repetitions. Here, $\{i_1, i_3\}$ (resp., $\{i_2, i_4\}$) can take any distinct two values in the range from the start to the end of $\mathbf{R}_1$, i.e., from 0 to $(m + 1)\gamma - 1$ (see also Fig. 5.12).

Regarding $F^2_{P_9,1}$, we need to account for Case 9.2, Case 9.3, and Case 9.4. For Case 9.2, we need to distinguish between two situations; when $r < e$ and when $r > e$, which gives the two summations of $\mathcal{B}_{P_9}$ in $F^2_{P_9,1}$. In Case 9.2, c_1 and c_4 each connects overlaps in $\mathbf{R}_r$ and $\mathbf{R}_e$, while c_2 and c_3 each connects overlaps in $\mathbf{R}_r$ only. For the first summation in $F^2_{P_9,1}$, $\mathcal{B}_{P_9}$ in (5.53) has $r = 1$ and $e = 2$. Thus, $\{i_1, i_4\}$ can take any distinct two values in the range from the start of $\mathbf{R}_2$ to the end of $\mathbf{R}_1$, i.e., from γ to $(m + 1)\gamma - 1$. Moreover, i_2 (resp., i_3) can take any value in the range from the start to the end of $\mathbf{R}_1$, i.e., from 0 to $(m + 1)\gamma - 1$. For the second summation, $\mathcal{B}_{P_9}$ in (5.53) has $r = 2$ and $e = 1$. Thus, $\{i_1, i_4\}$ can take any distinct two values in the range from the start of $\mathbf{R}_2$ to the end of $\mathbf{R}_1$, i.e., from 0 to $m\gamma - 1$. Moreover, i_2 (resp., i_3) can take any value in the range from the start to the end of $\mathbf{R}_2$, i.e., from 0 to $(m + 1)\gamma - 1$. Note that the ranges of i_2 and i_3 are the same in Case 9.2. However, i_2 and i_3 still need to be separated in order to count all the ways of forming P_9. The above distinction is not needed for neither Case 9.3 nor Case 9.4 since the two replicas have the same number of degree-2 overlaps with similar connectivity. In Case 9.3, and as shown in Fig. 5.12, c_1 and c_3 each connects overlaps in $\mathbf{R}_r$ and $\mathbf{R}_e$, while c_2 (resp., c_4) connects overlaps in $\mathbf{R}_r$ (resp., $\mathbf{R}_e$) only. For the third summation in $F^2_{P_9,1}$, $\mathcal{C}_{P_9}$ in (5.54) has $r = 1$ and $e = 2$. Thus, $\{i_1, i_3\}$ can take any distinct two values in the range from the start of $\mathbf{R}_2$ to the end of $\mathbf{R}_1$, i.e., from γ to $(m + 1)\gamma - 1$. Moreover, i_2 (resp., i_4) can take any value in the range from the start to the end of $\mathbf{R}_1$ (resp., $\mathbf{R}_2$), i.e., from 0 (resp., γ) to $(m + 1)\gamma - 1$ (resp., $(m + 2)\gamma - 1$). Note that the ranges of i_2 and i_4 are different in Case 9.3. In Case 9.4, all the CNs connect overlaps in $\mathbf{R}_r$ and $\mathbf{R}_e$. For the fourth summation in $F^2_{P_9,1}$, $\mathcal{D}_{P_9}$ in (5.55) has $r = 1$ and $e = 2$. Thus, $\{i_1, i_4\}$ can take any distinct two values in the range from the start of $\mathbf{R}_2$ to the end of $\mathbf{R}_1$, i.e., from γ to $(m+1)\gamma-1$. Moreover, i_2 (resp., i_3) can take any value in the range from the start of $\mathbf{R}_2$ to the end of $\mathbf{R}_1$, i.e., from γ to $(m+1)\gamma-1$. Note that the ranges of i_2 and i_3 are the same in Case 9.4, but they are still separated in order to count all the ways of forming P_9. Moreover, the multiplication by $\frac{1}{2}$ for the counts of $\mathcal{D}_{P_9}$ is also to account for repetitions (as with $\mathcal{A}_{P_9}$).

As for $F^3_{P_9,1}$, the overlaps can be in 2 replicas (the first four summations in $F^3_{P_9,1}$) or 3 replicas (the following six summations in $F^3_{P_9,1}$). The first four summations are derived in a way similar to what we did for $F^2_{P_9,1}$, with a change in the summation indices; $\mathbf{R}_2$ is replaced by $\mathbf{R}_3$ here. Then, we need to account for Case 9.5 (fifth to seventh summations) and Case 9.6 (eighth to tenth summations). In Case 9.5, c_1 (resp., c_2) connects overlaps in $\mathbf{R}_r$ and $\mathbf{R}_s$ (resp., $\mathbf{R}_r$ only). On the other hand, c_3 (resp., c_4) connects overlaps in $\mathbf{R}_r$ and $\mathbf{R}_e$ (resp., $\mathbf{R}_e$ and $\mathbf{R}_s$). For the fifth summation, $\mathcal{E}_{P_9}$ in (5.56) has $r=1$, $e=2$, and $s=3$. Thus, i_1 (resp., i_2, i_3, and i_4) can take any value in the range from the start of $\mathbf{R}_3$ (resp., $\mathbf{R}_1$, $\mathbf{R}_2$, and $\mathbf{R}_3$) to the end of $\mathbf{R}_1$ (resp., $\mathbf{R}_1$, $\mathbf{R}_1$, and $\mathbf{R}_2$), i.e., from 2γ (resp., 0, γ, and 2γ) to $(m+1)\gamma-1$ (resp., $(m+1)\gamma-1$, $(m+1)\gamma-1$, and $(m+2)\gamma-1$). For the sixth summation, $\mathcal{E}_{P_9}$ in (5.56) has $r=2$, $e=1$, and $s=3$. Thus, i_1 (resp., i_2, i_3, and i_4) can take any value in the range from the start of $\mathbf{R}_3$ (resp., $\mathbf{R}_2$, $\mathbf{R}_2$, $\mathbf{R}_3$) to the end of $\mathbf{R}_2$ (resp., $\mathbf{R}_2$, $\mathbf{R}_1$, and $\mathbf{R}_1$), i.e., from γ (resp., 0, 0, and γ) to $(m+1)\gamma-1$ (resp., $(m+1)\gamma-1$, $m\gamma-1$, and $m\gamma-1$). For the seventh summation, $\mathcal{E}_{P_9}$ in (5.56) has $r=3$, $e=1$, and $s=2$. Thus, i_1 (resp., i_2, i_3, and i_4) can take any value in the range from the start of $\mathbf{R}_3$ (resp., $\mathbf{R}_3$, $\mathbf{R}_3$, and $\mathbf{R}_2$) to the end of $\mathbf{R}_2$ (resp., $\mathbf{R}_3$, $\mathbf{R}_1$, and $\mathbf{R}_1$), i.e., from 0 (resp., 0, 0, and $-\gamma$) to $m\gamma-1$ (resp., $(m+1)\gamma-1$, $(m-1)\gamma-1$, and $(m-1)\gamma-1$). In Case 9.6, and as shown in Fig. 5.12, c_1 and c_4 each connects overlaps in $\mathbf{R}_r$ and $\mathbf{R}_s$. On the other hand, c_2 and c_3 each connects overlaps in $\mathbf{R}_r$ and $\mathbf{R}_e$. For the eighth summation, $\mathcal{G}_{P_9}$ in (5.57) has $r=1$, $e=2$, and $s=3$. Thus, $\{i_1, i_4\}$ can take any distinct two values in the range from the start of $\mathbf{R}_3$ to the end of $\mathbf{R}_1$, i.e., from 2γ to $(m+1)\gamma-1$. Moreover, i_2 (resp., i_3) can take any value in the range from the start of $\mathbf{R}_2$ to the end of $\mathbf{R}_1$, i.e., from γ to $(m+1)\gamma-1$. For the ninth summation, $\mathcal{G}_{P_9}$ in (5.57) has $r = 2$, $e = 1$, and $s = 3$ (see Fig. 5.12). Thus, $\{i_1, i_4\}$ can take any distinct two values in the range from the start of $\mathbf{R}_3$ to the end of $\mathbf{R}_2$, i.e., from γ to $(m + 1)\gamma - 1$. Moreover, i_2 (resp., i_3) can take any value in the range from the start of $\mathbf{R}_2$ to the end of $\mathbf{R}_1$, i.e., from 0 to $m\gamma - 1$. For the tenth summation, $\mathcal{G}_{P_9}$ in (5.57) has $r = 3$, $e = 1$, and $s = 2$. Thus, $\{i_1, i_4\}$ can take any distinct two values in the range from the start of $\mathbf{R}_3$ to the end of $\mathbf{R}_2$, i.e., from 0 to $m\gamma - 1$. Moreover, i_2 (resp., i_3) can take any value in the range from the start of $\mathbf{R}_3$ to the end of $\mathbf{R}_1$, i.e., from 0 to $(m - 1)\gamma - 1$.

Regarding $F^{k\geqslant 4}_{P_9,1}$, the overlaps can be in 2 replicas (the first four summations in $F^{k\geqslant 4}_{P_9,1}$), 3 replicas (the following six summations in $F^{k\geqslant 4}_{P_9,1}$), or 4 replicas (the last three summations in $F^{k\geqslant 4}_{P_9,1}$). The first four summations are derived in a way similar to what we did for $F^{2}_{P_9,1}$, with a change in the summation indices; $\mathbf{R}_2$ is replaced by $\mathbf{R}_k$. The following six summations are derived in a way similar to what we did for $F^{3}_{P_9,1}$, with a change in the summation indices; $\mathbf{R}_2$ and $\mathbf{R}_3$ are replaced by $\mathbf{R}_h$ and $\mathbf{R}_k$, respectively, which also requires changing these six summations of $\mathcal{E}_{P_9}$ and $\mathcal{G}_{P_9}$ to be double summations. The following three summations are associated with Case 9.7. In Case 9.7, c_1 (resp., c_2) connects overlaps in $\mathbf{R}_r$ and $\mathbf{R}_u$ (resp., $\mathbf{R}_r$ and $\mathbf{R}_e$). On the other hand, c_3 (resp., c_4) connects overlaps in $\mathbf{R}_e$ and $\mathbf{R}_s$ (resp., $\mathbf{R}_s$ and $\mathbf{R}_u$). See Fig. 5.12 for more illustration. The two overlaps connected to the $c_1 - c_2$ overlap through CNs are the $c_2 - c_3$ and the $c_1 - c_4$ overlaps. There are three situations to distinguish between; these two overlaps are in the second and last replicas, in the third and last replicas, and in the second and third replicas. The ordering of replicas here is with respect to the four replicas in which the overlaps of P_9 exist. For the eleventh (triple) summation, $\mathcal{I}_{P_9}$ in (5.58) has $r = 1$, $e = h$, $s = w$, and $u = k$ (see Fig. 5.12). Thus, i_1 (resp., i_2, i_3, and i_4) can take any value in the range from the start of $\mathbf{R}_k$ (resp., $\mathbf{R}_h$, $\mathbf{R}_w$, and $\mathbf{R}_k$) to the end of $\mathbf{R}_1$ (resp., $\mathbf{R}_1$, $\mathbf{R}_h$, and $\mathbf{R}_w$), i.e., from $(k - 1)\gamma$ (resp., $(h - 1)\gamma$, $(w - 1)\gamma$, and $(k - 1)\gamma$) to $(m + 1)\gamma - 1$ (resp., $(m + 1)\gamma - 1$, $(m + h)\gamma - 1$, and $(m + w)\gamma - 1$). For the twelfth (triple) summation, $\mathcal{I}_{P_9}$ in (5.58) has $r = 1$, $e = w$, $s = h$, and $u = k$. Thus, i_1 (resp., i_2, i_3, and i_4) can take any value in the range from the start of $\mathbf{R}_k$ (resp., $\mathbf{R}_w$, $\mathbf{R}_w$, and $\mathbf{R}_k$) to the end of $\mathbf{R}_1$ (resp., $\mathbf{R}_1$, $\mathbf{R}_h$, and $\mathbf{R}_h$), i.e., from $(k-1)\gamma$ (resp., $(w-1)\gamma$, $(w-1)\gamma$, and $(k-1)\gamma$) to $(m+1)\gamma-1$ (resp., $(m+1)\gamma-1$, $(m+h)\gamma-1$, and $(m+h)\gamma-1$). For the thirteenth (triple) summation, $\mathcal{I}_{P_9}$ in (5.58) has $r=1$, $e=h$, $s=k$, and $u=w$. Thus, i_1 (resp., i_2, i_3, and i_4) can take any value in the range from the start of $\mathbf{R}_w$ (resp., $\mathbf{R}_h$, $\mathbf{R}_k$, and $\mathbf{R}_k$) to the end of $\mathbf{R}_1$ (resp., $\mathbf{R}_1$, $\mathbf{R}_h$, and $\mathbf{R}_w$), i.e., from $(w-1)\gamma$ (resp., $(h-1)\gamma$, $(k-1)\gamma$, and $(k-1)\gamma$) to $(m+1)\gamma-1$ (resp., $(m+1)\gamma-1$, $(m+h)\gamma-1$, and $(m+w)\gamma-1$). The outer two summations are over all possible values of h and w, and we have $1<h<k-1$ and $h<w<k$ (similar to Patterns P_4 and P_7).

Note that c_1 and c_3 are not adjacent in P_9, and the same applies for c_2 and c_4. Thus, it is possible to have $\overline{i_1}=\overline{i_3}$ and $\overline{i_2}=\overline{i_4}$, but not $i_1=i_3$ nor $i_2=i_4$, for that pattern. $\square$

CHAPTER 6

Conclusion

6.1 Summary of Our Results

The principal focus of this dissertation was the analysis and design of high performance graph-based codes for modern dense storage devices. In particular, this dissertation studied three main problems. The first problem was about predicting the performance of an NB-LDPC code in magnetic recording systems. The second problem was about optimizing NB-LDPC codes via adjusting their edge weights in order to improve the performance over practical storage channels. The third problem was about designing high performance SC codes through optimal partitioning and enhanced lifting.

Regarding the first problem, we demonstrated that the nature of the detrimental objects that dominate the error floor region of an NB-LDPC code critically depends on the channel of interest. These detrimental objects are subgraphs with certain properties in the graph of the NB-LDPC code. We introduced BASs and FOBASs to capture the dominant objects in the case of magnetic recording channels. We then developed a linear-algebraic method to predict the error floor performance of an NB-LDPC code based on the dominant FOBASs in the graph of the code. Our method was demonstrated to not only accurately predict the error floor performance of NB-LDPC codes, but also perform this task orders of magnitude faster than the traditional MC simulations.

Regarding the second problem, we showed how asymmetry, as in practical Flash channels, affects the dominant objects of NB-LDPC codes. Consequently, we introduced finer definitions of absorbing sets to capture these dominant objects and precisely distinguish between their topologies. In particular, we defined GASs and GASTs. Then, we proposed a matrix theoretic representation of GASTs; we expressed a GAST as a set of submatrices, which we call WCMs. By forcing the null spaces of its WCMs to have a certain property, we demonstrated that a GAST is provably removed from the graph of the NB-LDPC code. Moreover, we clarified how the WCM framework can be customized to optimize NB-LDPC codes for magnetic recording devices as we all Flash memories. Simulation results showed gains in orders of magnitude compared to prior state-of-the-art.

Next, we provided an in-depth analysis and extensions of the WCM framework. We proved the optimality of the WCM framework and illustrated the significant reduction in the number of matrices it operates on compared to suboptimal ideas. We also provided a study of the null spaces of WCMs. Furthermore, we presented the minimum number of edge weight changes needed to remove a GAST in addition to how to choose these changes. Then, we presented a new class of detrimental objects, OSTs, which are the secondary cause of the error floor in NB-LDPC codes with even column weights. We extended the repertoire of the WCM framework by showing its benefits for NB graph-based codes, including SC codes, with different parameters and designed for different applications.

Regarding the third problem, we derived a channel-aware combinatorial approach to design high performance binary and non-binary SC codes. We first identified a common substructure which exists in multiple detrimental configurations. Moreover, we introduced the concept of the pattern to capture the configurations in the protograph that can result in instances of the common substructure of interest after lifting. In the OO stage of the approach, we derive a discrete optimization problem, in which we express the total number of patterns in terms of the overlap parameters that characterize the partitioning. Then, we solve this optimization problem to determine the partitioning. In the CPO stage of the approach, we optimize the circulant powers in order to further reduce the number of common substructure instances in the lifted graph.

Simulation results demonstrated that by optimally exploiting the degree of freedom guaranteed via partitioning in the design of SC codes, these codes not only outperform prior state-of-the-art counterparts by orders of magnitude, but also can outperform block codes of similar parameters. While our main focus in the chapter where we studied the third problem was on magnetic recording systems, we already demonstrated the significant gains offered by the OO-CPO approach for Flash memory systems in [38].

6.2 Future Directions

As for the error floor prediction, we deployed bit-based detection in the magnetic recording system used here. One future direction is to extend the analysis to the case of symbol-based detection since such detection has its advantages in systems incorporating intrinsic memory. Another interesting direction is to develop the error floor prediction method for asymmetric Flash channels. Particularly, the NLM Flash channel will be an intriguing option in this regard. A third possible direction is to derive error floor prediction methods for multi-dimensional (MD) storage devices, e.g., TDMR and 3-D Flash devices.

As for the NB-LDPC code optimization, the WCM framework currently works for NB-LDPC codes with fixed column weights (regular VN degrees). One future direction is to extend the WCM framework such that it can optimize NB-LDPC codes with irregular VN degrees since these NB-LDPC codes are also used in many applications. Another promising direction is to study the NB-LDPC code optimization problem in MD storage systems and derive the necessary additions over the WCM framework.

As for the design of high performance SC codes, the SC codes we investigated here also have fixed column weights (regular VN degrees). One future direction is to study SC codes with irregular VN degrees asymptotically and optimize the degree distribution of an irregular SC code. Another intriguing direction is to design high performance time-variant SC codes, where the non-zero parts of different replicas are not the same. These two future directions may also be combined together. A third direction is to search for theoretical techniques to further speed up the OO-CPO approach.

Multi-dimensional SC (MD-SC) codes are more robust against burst erasures and MD channel non-uniformity, and they have improved iterative decoding thresholds compared to 1D-SC codes [88, 89, 90, 91]. Recently in [46], we presented some promising results about MD-SC codes. Our focus was to design MD-SC codes with low population of short cycles through informed relocation of circulants. We derived the theoretical condition on a cycle of a particular length to stay active (same length) in the graph of an MD-SC code. Our goal was then to break this condition for as many cycles as possible via relocations. We developed an algorithm that decides concerning the relocation of a specific circulant based on the majority of the votes from all the cycles in which this circulant exists. We presented simulation results showing the significant performance advantage of our MD-SC codes compared to 1D-SC codes with similar parameters [46].

The technique we proposed in [46] to design MD-SC codes is for the canonical AWGN channel. One future direction is to extend the analysis to practical MD storage channels, e.g., the TDMR channel. Another direction is to propose improved windowed decoding algorithms customized for MD-SC codes. A third interesting direction is to propose MD codes that are based on parity-check constraints derived from the impulse response of the MD channel for TDMR and 3-D Flash channels.

These are all future directions that focus mostly on data storage applications. Additionally, there are other applications in which graph-based codes can significantly increase the robustness of the system. One hot application is the distributed computations problem, where graph-based codes with certain properties can be used to overcome the issue of straggling computing nodes. Other applications of interest include privacy/secrecy, distributed storage, machine learning, in addition to wireless communications.

References

[1] R. G. Gallager, *Low-Density Parity-Check Codes.* Cambridge, MA: MIT Press, 1963.

[2] M. Davey and D. MacKay, "Low-density parity-check codes over GF(q)," *IEEE Commun. Lett.*, vol. 2, no. 6, pp. 165–167, Jun. 1998.

[3] L. Dolecek, Z. Zhang, V. Anantharam, M. Wainwright, and B. Nikolic, "Analysis of absorbing sets and fully absorbing sets of array-based LDPC codes," *IEEE Trans. Inf. Theory*, vol. 56, no. 1, pp. 181–201, Jan. 2010.

[4] J. Wang, L. Dolecek, and R. Wesel, "The cycle consistency matrix approach to absorbing sets in separable circulant-based LDPC codes," *IEEE Trans. Inf. Theory*, vol. 59, no. 4, pp. 2293–2314, Apr. 2013.

[5] O. Milenkovic, E. Soljanin, and P. Whiting, "Asymptotic spectra of trapping sets in regular and irregular LDPC code ensembles," *IEEE Trans. Inf. Theory*, vol. 53, no. 1, pp. 39–55, Jan. 2007.

[6] M. Karimi and A. H. Banihashemi, "Efficient algorithm for finding dominant trapping sets of LDPC codes," *IEEE Trans. Inf. Theory*, vol. 58, no. 11, pp. 6942–6958, Nov. 2012.

[7] C. Poulliat, M. Fossorier, and D. Declercq, "Design of regular $(2, d_c)$-LDPC codes over GF(q) using their binary images," *IEEE Trans. Commun.*, vol. 56, no. 10, pp. 1626–1635, Oct. 2008.

[8] L. Dolecek, D. Divsalar, Y. Sun, and B. Amiri, “Non-binary protograph-based LDPC codes: enumerators, analysis, and designs,” *IEEE Trans. Inf. Theory*, vol. 60, no. 7, pp. 3913–3941, Jul. 2014.

[9] A. Bazarsky, N. Presman, and S. Litsyn, “Design of non-binary quasi-cyclic LDPC codes by ACE optimization,” in *Proc. IEEE Inf. Theory Workshop (ITW)*, Sevilla, Spain, Sep. 2013, pp. 1–5.

[10] B. Amiri, J. Kliewer, and L. Dolecek, “Analysis and enumeration of absorbing sets for non-binary graph-based codes,” *IEEE Trans. Commun.*, vol. 62, no. 2, pp. 398–409, Feb. 2014.

[11] B. Vasic and E. Kurtas, *Coding and Signal Processing for Magnetic Recording Systems.* CRC Press, 2005.

[12] G. Colavolpe and G. Germi, “On the application of factor graphs and the sum-product algorithm to ISI channels,” *IEEE Trans. Commun.*, vol. 53, no. 5, pp. 818–825, May 2005.

[13] Y. Fang, P. Chen, L. Wang, and F. Lau, “Design of protograph LDPC codes for partial response channels,” *IEEE Trans. Commun.*, vol. 60, no. 10, pp. 2809–2819, Oct. 2012.

[14] Y. Han and W. Ryan, “Low-floor detection/decoding of LDPC-coded partial response channels,” *IEEE J. Sel. Areas Commun.*, vol. 28, no. 2, pp. 252–260, Feb. 2010.

[15] H. Zhong, T. Zhong, and E. Haratsch, “Quasi-cyclic LDPC codes for the magnetic recording channel: code design and VLSI implementation,” *IEEE Trans. Magn.*, vol. 43, no. 3, pp. 1118–1123 , Mar. 2007.

[16] S. Jeon and B. Kumar, “Binary SOVA and nonbinary LDPC codes for turbo equalization in magnetic recording channels,” *IEEE Trans. Magn.*, vol. 46, no. 6, pp. 2248–2251, June 2010.

[17] A. Risso, “Layered LDPC decoding over GF(q) for magnetic recording channel,” *IEEE Trans. Magn.*, vol. 45, no. 10, pp. 3683–3686 , Oct. 2009.

[18] C. A. Cole, S. G. Wilson, E. K. Hall, and T. R. Giallorenzi, “A general method for finding low error rates of LDPC codes,” May 2006. [Online]. Available: http://arxiv.org/abs/cs/0605051

[19] E. Cavus, C. Haymes, B. Daneshrad, “An IS simulation technique for very low BER performance evaluation of LDPC codes,” in *Proc. IEEE Int. Conf. Commun. (ICC)*, Istanbul, Turkey, June 2006, pp. 1095–1100.

[20] L. Dolecek, P. Lee, Z. Zhang, V. Anantharam, B. Nikolic, and M. Wainwright, “Predicting error floors of structured LDPC codes: deterministic bounds and estimates,” *IEEE J. Sel. Areas Commun.*, vol. 27, no. 6, pp. 908–917, Aug. 2009.

[21] H. Xiao, A. Banihashemi, and M. Karimi, “Error rate estimation of low-density parity-check codes decoded by quantized soft-decision iterative algorithms,” *IEEE Trans. Commun.*, vol. 61, no. 2, pp. 474–484, Feb. 2013.

[22] X. Hu, Z. Li, B. V. K. V. Kumar, and R. Barndt, “Error floor estimation of long LDPC codes on partial response channels,” in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Washington, DC, USA, Dec. 2007, pp. 259–264.

[23] X. Hu, Z. Li, B. V. K. V. Kumar, and R. Barndt, “Error floor estimation of long LDPC codes on magnetic recording channels,” *IEEE Trans. Magn.*, vol. 46, no. 6, pp. 1836–1839, Jun. 2010.

[24] T. Parnell, N. Papandreou, T. Mittelholzer, and H. Pozidis, “Modelling of the threshold voltage distributions of sub-20nm NAND flash memory,” in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Austin, TX, USA, Dec. 2014, pp. 2351–2356.

[25] Y. Cai, E. Haratsch, O. Mutlu, and K. Mai, "Threshold voltage distribution in MLC NAND flash memory: Characterization, analysis, and modeling," in *Proc. Design, Autom., Test Eur. Conf. Exhibition (DATE)*, Grenoble, France, Mar. 2013, pp. 1285–1290.

[26] Y. Maeda and H. Kaneko, "Error control coding for multilevel cell Flash memories using nonbinary low-density parity-check codes," in *Proc. 24th IEEE Int. Symp. Defect and Fault Tolerance in VLSI Systems (DFT)*, Chicago, IL, USA, Oct. 2009, pp. 367–375.

[27] K. Ho, C. Chen, and H. Chang, "A 520k (18900, 17010) array dispersion LDPC decoder architectures for NAND Flash memory," *IEEE Trans. VLSI Systems*, vol. 24, no. 4, pp. 1293–1304, Apr. 2016.

[28] J. Wang, K. Vakilinia, T.-Y. Chen, T. Courtade, G. Dong, T. Zhang, H. Shankar, and R. Wesel, "Enhanced precision through multiple reads for LDPC decoding in flash memories," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 5, pp. 880–891, May 2014.

[29] A. J. Felstrom and K. S. Zigangirov, "Time-varying periodic convolutional codes with low-density parity-check matrix," *IEEE Trans. Inf. Theory*, vol. 45, no. 6, pp. 2181–2191, Sep. 1999.

[30] A. E. Pusane, R. Smarandache, P. O. Vontobel, and D. J. Costello, "Deriving good LDPC convolutional codes from LDPC block codes," *IEEE Trans. Inf. Theory*, vol. 57, no. 2, pp. 835–857, Feb. 2011.

[31] S. Kudekar, T. J. Richardson, and R. L. Urbanke, "Spatially coupled ensembles universally achieve capacity under belief propagation," *IEEE Trans. Inf. Theory*, vol. 59, no. 12, pp. 7761–7813, Dec. 2013.

[32] H. Esfahanizadeh, A. Hareedy, and L. Dolecek, "Spatially-coupled codes optimized for magnetic recording applications," *IEEE Trans. Magn.*, vol. 53, no. 2, pp. 1–11, Feb. 2017.

[33] A. R. Iyengar, M. Papaleo, P. H. Siegel, J. K. Wolf, A. Vanelli-Coralli, and G. E. Corazza, "Windowed decoding of protograph-based LDPC convolutional codes over erasure channels," *IEEE Trans. Inf. Theory*, vol. 58, no. 4, pp. 2303–2320, Apr. 2012.

[34] H. Esfahanizadeh, A. Hareedy, and L. Dolecek, "A novel combinatorial framework to construct spatially-coupled codes: minimum overlap partitioning," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Aachen, Germany, Jun. 2017, pp. 1693–1697.

[35] H. Esfahanizadeh, A. Hareedy, R. Wu, R. Galbraith, and L. Dolecek, "Spatially-coupled codes for channels with SNR variation," accepted at *IEEE Trans. Magn.*, doi: 10.1109/TMAG.2018.2853087, Jun. 2018.

[36] D. G. M. Mitchell and E. Rosnes, "Edge spreading design of high rate array-based SC-LDPC codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Aachen, Germany, Jun. 2017, pp. 2940–2944.

[37] H. Esfahanizadeh, A. Hareedy, and L. Dolecek, "Finite-length construction of high performance spatially-coupled codes via optimized partitioning and lifting," accepted at *IEEE Trans. Commun.*, doi: 10.1109/TCOMM.2018.2867493, Aug. 2018.

[38] A. Hareedy, H. Esfahanizadeh, and L. Dolecek, "High performance non-binary spatially-coupled codes for flash memories," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Kaohsiung, Taiwan, Nov. 2017, pp. 229–233.

[39] A. Hareedy, B. Amiri, R. Galbraith, S. Zhao, and L. Dolecek, "Non-binary LDPC code optimization for partial-response channels," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, San Diego, CA, USA, Dec. 2015, pp. 1–6.

[40] A. Hareedy, B. Amiri, R. Galbraith, and L. Dolecek, "Non-binary LDPC codes for magnetic recording channels: error floor analysis and optimized code design," *IEEE Trans. Commun.*, vol. 64, no. 8, pp. 3194–3207, Aug. 2016.

[41] A. Hareedy, C. Lanka, C. Schoeny, and L. Dolecek, "The weight consistency matrix framework for general non-binary LDPC code optimization: applications in Flash memories," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Barcelona, Spain, Jul. 2016, pp. 2709–2713.

[42] A. Hareedy, C. Lanka, and L. Dolecek, "A general non-binary LDPC code optimization framework suitable for dense Flash memory and magnetic storage," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 9, pp. 2402–2415, Sep. 2016.

[43] A. Hareedy, C. Lanka, N. Guo, and L. Dolecek, "A combinatorial methodology for optimizing non-binary graph-based codes: theoretical analysis and applications in data storage," accepted at *IEEE Trans. Inf. Theory*, doi: 10.1109/TIT.2018.2870437, Sep. 2018.

[44] A. Hareedy, H. Esfahanizadeh, A. Tan, and L. Dolecek, "Spatially-coupled code design for partial-response channels: optimal object-minimization approach," accepted at *IEEE Global Commun. Conf. (GLOBECOM)*, Abu Dhabi, UAE, Dec. 2018. [Online]. Available: http://arxiv.org/abs/1804.05504

[45] A. Hareedy, R. Wu, and L. Dolecek, "A channel-aware combinatorial approach to design high performance spatially-coupled codes for magnetic recording systems," Sep. 2018. [Online]. Available: http://www.uclacodess.org/papers/AHH_OOCPO_TIT.pdf

[46] H. Esfahanizadeh, A. Hareedy, and L. Dolecek, "Multi-dimensional spatially-coupled code design through informed relocation of circulants," accepted at *56th Annual Allerton Conf. Commun., Control, and Computing*, Monticello, IL, USA, Oct. 2018. [Online]. Available: http://arxiv.org/abs/1809.04798

[47] A. Hareedy, B. Amiri, R. Galbraith, and L. Dolecek, "Supplementary results on non-binary LDPC codes for partial-response channels," Jun. 2016. [Online]. Available: http://www.uclacodess.org/papers/supp_results_bas.pdf.

[48] H. Esfahanizadeh, A. Hareedy and L. Dolecek, "The finite length analysis of spatially-coupled codes for 1-D magnetic recording channels," in *50th Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA, Nov. 2016, pp. 1128–1132.

[49] Y. Cai, G. Yalcin, O. Mutlu, E. Haratsch, A. Cristal, O. Unsal, and K. Mai, "Flash correct-and-refresh: Retention-aware error management for increased Flash memory lifetime," in *Proc. IEEE 30th IEEE Int. Conf. Comput. Des. (ICCD)*, Montreal, Quebec, Canada, Oct. 2012, pp. 94–101.

[50] T. Souvignier, M. Öberg, P. Siegel, R. Swanson, and J. Wolf, "Turbo decoding for partial response channels," *IEEE Trans. Commun.*, vol. 48, no. 8, pp. 1297–1308, Aug. 2000.

[51] S. Srinivasa, Y. Chen, and S. Dahandeh, "A communication-theoretic framework for 2-DMR channel modeling: performance evaluation of coding and signal processing methods," *IEEE Trans. Magn.*, vol. 50, no. 3, pp. 6–12, Mar. 2014.

[52] D. Declercq and M. Fossorier, "Decoding algorithms for nonbinary LDPC codes over GF(q)," *IEEE Trans. Commun.*, vol. 55, no. 4, pp. 633–643, Apr. 2007.

[53] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inf. Theory*, vol. 20, pp. 284–287, Mar. 1974.

[54] J. Moon and J. Park, "Pattern-dependent noise prediction in signal dependent noise," *IEEE J. Sel. Areas Commun.*, vol. 19, no. 4, pp. 730–743 , Apr. 2001.

[55] B. Amiri, C. Lin, and L. Dolecek, "Asymptotic distribution of absorbing sets and fully absorbing sets for regular sparse code ensembles," *IEEE Trans. Commun.*, vol. 61, no. 2, pp. 455–464, Feb. 2013.

[56] T. Oenning and J. Moon, "A low-density generator matrix interpretation of parallel concatenated single bit parity codes," *IEEE Trans. Magn.*, vol. 37, no. 2, pp. 737–741, Mar. 2001.

[57] T. Duman and E. Kurtas, "Comprehensive performance investigation of turbo codes over high density magnetic recording channels," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Rio de Janeiro, Brazil, Dec. 1999, pp. 744–748.

[58] M. Karimi and A. Banihashemi, "On characterization of elementary trapping sets of variable-regular LDPC codes," *IEEE Trans. Inf. Theory*, vol. 60, no. 9, pp. 5188–5203, Sept. 2014.

[59] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 599–618, Feb. 2001.

[60] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 619–637, Feb. 2001.

[61] A. H. Hareedy and M. M. Khairy, "Selective max-min algorithm for low-density parity-check decoding," *IET Commun.*, vol. 7, no. 1, pp. 65–70, Jan. 2013.

[62] C. A. Aslam, Y. L. Guang, and K. Cai, "Read and write voltage signal optimization for multi-level-cell (MLC) NAND Flash memory," *IEEE Trans. Commun.*, vol. 64, no. 4, pp. 1613–1623, Apr. 2016.

[63] R. Cohen and Y. Cassuto, "Iterative decoding of LDPC codes over the q-ary partial erasure channel," *IEEE Trans. Inf. Theory*, vol. 62, no. 5, pp. 2658–2672, May 2016.

[64] Y. Cassuto and A. Shokrollahi, "LDPC codes for 2D arrays," *IEEE Trans. Inf. Theory*, vol. 60, no. 6, pp. 3279–3291, Jun. 2014.

[65] A. Tomasoni, S. Bellini, and M. Ferrari, "Thresholds of absorbing sets in low-density parity-check codes," *IEEE Trans. Commun.*, vol. 65, no. 8, pp. 3238–3249, Aug. 2017.

[66] M. Ivkovic, S. K. Chilappagari, and B. Vasic, "Eliminating trapping sets in low-density parity-check codes by using Tanner graph covers," *IEEE Trans. Inf. Theory*, vol. 54, no. 8, pp. 3763–3768, Aug. 2008.

[67] A. McGregor and O. Milenkovic, "On the Hardness of Approximating Stopping and Trapping Sets," *IEEE Trans. Inf. Theory*, vol. 56, no. 4, pp. 1640–1650, Apr. 2010.

[68] B. K. Butler and P. H. Siegel, "Error floor approximation for LDPC codes in the AWGN channel," *IEEE Trans. Inf. Theory*, vol. 60, no. 12, pp. 7416–7441, Dec. 2014.

[69] D. V. Nguyen, S. K. Chilappagari, M. W. Marcellin, and B. Vasic, "On the construction of structured LDPC codes free of small trapping sets," *IEEE Trans. Inf. Theory*, vol. 58, no. 4, pp. 2280–2302, Apr. 2012.

[70] Q. Diao, Y. Y. Tai, S. Lin, and K. Abdel-Ghaffar, "LDPC codes on partial geometries: construction, trapping set structure, and puncturing," *IEEE Trans. Inf. Theory*, vol. 59, no. 12, pp. 7898–7914, Dec. 2013.

[71] Q. Huang, Q. Diao, S. Lin, and K. Abdel-Ghaffar, "Cyclic and quasi-cyclic LDPC codes on constrained parity-check matrices and their trapping sets," *IEEE Trans. Inf. Theory*, vol. 58, no. 5, pp. 2648–2671, May 2012.

[72] M. Lentmaier, A. Sridharan, D. J. Costello, and K. S. Zigangirov, "Iterative decoding threshold analysis for LDPC convolutional codes," *IEEE Trans. Inf. Theory*, vol. 56, no. 10, pp. 5274–5289, Oct. 2010.

[73] I. Andriyanova and A. Graell i Amat, "Threshold saturation for nonbinary SC-LDPC codes on the binary erasure channel," *IEEE Trans. Inf. Theory*, vol. 62, no. 5, pp. 2622–2638, May 2016.

[74] P. M. Olmos and R. L. Urbanke, "A scaling law to predict the finite-length performance of spatially-coupled LDPC codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 6, pp. 3164–3184, Jun. 2015.

[75] Y. Toriyama and D. Markovic, "A 2.267-Gb/s, 93.7-pJ/bit non-binary LDPC decoder with logarithmic quantization and dual-decoding algorithm scheme for storage applications," *IEEE J. Solid-State Circuits*, vol. 53, no. 8, pp. 2378–2388, Aug. 2018.

[76] N. Ul Hassan, M. Lentmaier, I. Andriyanova, and G. P. Fettweis, "Improving code diversity on block-fading channels by spatial coupling," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Honolulu, HI, USA, Jun. 2014, pp. 2311–2315.

[77] A. Piemontese, A. Graell i Amat, and G. Colavolpe, "Nonbinary spatially-coupled LDPC codes on the binary erasure channel," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Budapest, Hungary, Jun. 2013, pp. 3270–3274.

[78] D. G. M. Mitchell, M. Lentmaier, and D. J. Costello, "Spatially coupled LDPC codes constructed from protographs," *IEEE Trans. Inf. Theory*, vol. 61, no. 9, pp. 4866–4889, Sep. 2015.

[79] Y. Xie, L. Yang, P. Kang, and J. Yuan, "Euclidean geometry-based spatially coupled LDPC codes for storage," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 9, pp. 2498–2509, Sep. 2016.

[80] D. G. M. Mitchell, L. Dolecek, and D. J. Costello, Jr., "Absorbing set characterization of array-based spatially coupled LDPC codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Honolulu, HI, USA, Jun. 2014, pp. 886–890.

[81] A. Beemer, S. Habib, C. A. Kelley, and J. Kliewer, "A generalized algebraic approach to optimizing SC-LDPC codes," in *Proc. 55th Annual Allerton Conf. Commun., Control, and Computing*, Monticello, IL, USA, Oct. 2017, pp. 672–679.

[82] I. E. Bocharova, B. D. Kudryashov, and R. Johannesson, “Searching for binary and nonbinary block and convolutional LDPC codes,” *IEEE Trans. Inf. Theory*, vol. 62, no. 1, pp. 163–183, Jan. 2016.

[83] W. Phakphisut, P. Supnithi, and N. Puttarak, “EXIT chart analysis of nonbinary protograph LDPC codes for partial response channels,” *IEEE Trans. Magn.*, vol. 50, no. 11, Nov. 2014, Art. no. 3101904.

[84] Z. Qin, K. Cai, Y. Ng, “Iterative detection and decoding for non-binary LDPC coded partial-response channels with written-in errors,” *IET Commun.*, vol. 10, no. 4, pp. 399–406 , Mar. 2016.

[85] P. Chen, C. Kui, L. Kong, Z. Chen, M. Zhang, “Non-binary protograph-based LDPC codes for 2-D-ISI magnetic recording channels,” *IEEE Trans. Magn.*, vol. 53, no. 11, Nov. 2017, Art. no. 8108905.

[86] S. Laendner and O. Milenkovic, “Algorithmic and combinatorial analysis of trapping sets in structured LDPC codes,” in *Proc. Int. Conf. Wireless Netw., Commun., and Mobile Comput.*, Maui, HI, Jun. 2005, pp. 630–635.

[87] M. P. C. Fossorier, “Quasi-cyclic low-density parity-check codes from circulant permutation matrices,” *IEEE Trans. Inf. Theory*, vol. 50, no. 8, pp. 1788–1793, Aug. 2004.

[88] D. Truhachev, D. G. M. Mitchell, M. Lentmaier, and D. J. Costello, “New codes on graphs constructed by connecting spatially coupled chains,” in *Proc. Inf. Theory and App. Workshop (ITA)*, Feb. 2012, pp. 392–397.

[89] R. Ohashi, K. Kasai, and K. Takeuchi, “Multi-dimensional spatially-coupled codes,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2013, pp. 2448–2452.

[90] L. Schmalen and K. Mahdaviani, "Laterally connected spatially coupled code chains for transmission over unstable parallel channels," in *Proc. Int. Symp. Turbo Codes Iterative Inf. Processing (ISTC)*, Aug. 2014, pp. 77–81.

[91] Y. Liu, Y. Li, and Y. Chi, "Spatially coupled LDPC codes constructed by parallelly connecting multiple chains," *IEEE Commun. Letters*, vol. 19, no. 9, pp. 1472–1475, Sep. 2015.