Title
Application-Driven Coding Techniques: From Cloud Storage to Quantum Communications

Author
Yang, Siyi

Publication Date
2021

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

Application-Driven Coding Techniques: From Cloud Storage to Quantum Communications

A thesis submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Electrical and Computer Engineering

by

Siyi Yang

2021

ABSTRACT OF THE THESIS

Application-Driven Coding Techniques:
From Cloud Storage to Quantum Communications

by

Siyi Yang

Doctor of Philosophy in Electrical and Computer Engineering
University of California, Los Angeles, 2021
Professor Lara Dolecek, Chair

Data-driven applications are becoming ubiquitous. This dissertation is focused on developing advanced channel coding techniques for improved reliability and latency in a variety of data-hungry applications, from cloud storage, to memory devices, and quantum communications.

The first line of our work focused on cloud storage. In order to accommodate the ever-growing data from various, possibly independent, sources and the dynamic nature of data usage rates in practical applications, modern cloud data storage systems are required to be scalable, flexible, and heterogeneous. The recent rise of the blockchain technology is also moving various information systems towards decentralization to achieve high privacy at low costs. We proposed channel codes with hierarchical locality that were the first to simultaneously achieve scalability and flexibility for both centralized cloud storage and decentralized storage networks (DSN). In particular, we proposed a joint coding scheme where each node receives extra protection through the cooperation with nodes in its neighborhood in a heterogeneous DSN with any given topology. Our proposed construction not only preserves desirable properties such as scalability and flexibility, which are critical in dynamic networks, but also adapts to arbitrary topologies, a property that is essential in DSNs but

has been overlooked in existing works.

The second line of our work focused on spatially-coupled (SC) codes design for advanced memory devices and quantum communications. SC codes have demonstrated potential in a variety of applications thanks to their excellent error-correcting performance and desirable structures that enable low latency decoding. While high memory SC codes are known to have superior performance, no prior work was able to produce practical codes due to computational complexity of the high-memory regime. We overcome this computational bottleneck in the finite-length construction of high-performance SC codes with high memory, with a novel coding framework that unifies seemingly disparate probabilistic and combinatorial approaches, and benefits from both. Simulation results show that codes obtained through our proposed method notably outperform state-of-the-art codes in a variety of practical settings, including flash memories and hard disk drives. Building on this new framework, we then developed a new class of channel codes for quantum communications. Combined with irregular-repeat-accumulate (IRA) codes that are known for excellent performance on low rate region, we constructed state-of-the-art SC-IRA codes for multidimensional quantum key distribution to efficiently generate private keys for one-time pad encrypted communications.

The thesis of Siyi Yang is approved.

Abeer Alwan

Richard D. Wesel

Gregory J. Pottie

Lara Dolecek, Committee Chair

University of California, Los Angeles

2021

To mom and dad.

Table of Contents

List of Figures

List of Tables

Acknowledgements

Foremost, I would like to express my sincere gratitude to my advisor Prof. Lara Dolecek for her continuous support of my study and research throughout my journey at LORIS Lab, where I have the chance to conduct state-of-the-art research topics in the field. Professor Dolecek has guided me to investigate various impactful research problems that are not only theoretically sound but also of practical importance. She has been open for different research topics and has her own strategic way to let me find out problems that are most suitable for me and beneficial for my future career. I also learned a lot from her experiences and expertises in presentation skills and writing styles, which definitely will affect my academic career deeply. In addition, Professor Dolecek has always been supportive and generous on recommending me in applications for various fellowships and future positions in academia.

I would also like to thank the rest of my thesis committee: Professor Abeer Alwan, Professor Gregory Pottie, and Professor Richard Wesel for their support and useful feedback on my defense and my thesis.

I am grateful to have excellent collaborators, including Dr. Ahmed Hareedy, Dr. Clayton Schoeny, Shyam Venkatasubramanian, and Murat Can Sarihan. Having a long-term collaboration with Ahmed has been the best decision I have made in the PhD journey. Let aside being responsible, supportive, and incredibly knowledgeable in every aspects, Ahmed has always been providing me with valuable and insightful suggestions not only on my research but also on decision making and career planning. I am looking forward to our continued collaboration after graduation. Clayton was the one who led me to my first research topic at LORIS, with whom I had my first journal paper published on TIT, which was definitely encouraging and meaningful for a newcomer in coding theory. I am also in debt to Shyam for his assistance in carrying out part of the simulations in my research, and Murat for providing experimental data for the interdisciplinary research on QKD. Apart from those I already have publications with, I also benefit from useful discussions with Dr. Homa Esfahanizadeh

and Ruiyi Wu when they were at LORIS.

Outside of UCLA, I have been lucky and honored to have the opportunity to know and collaborate with Professor Robert Calderbank. I am looking forward to continuing work with him on quantum error correction at Duke University. I also had the chance to work with Professor Chee Wei Wong and Professor Emina Soljanin, and learned a lot from them in the interdisciplinary collaboration on the QKD project. Apart from thesis-related research works, I have benefited from the experience as an intern at Intel Corporation, under manager Dr. Ravi Motwani and mentor Dr. Santhosh Kumar Vanaparthy. Both of them are experts in coding theory and my experience at Intel has been an amazing starting point of my work in LDPC codes. Additionally, I would also like to thank my mentors at Tsinghua University, Professor Yuan Shen and Professor Linglong Dai, and Professor Xiaodong Wang at Columbia University, without whose recommendations I would not have the opportunity to start the PhD journey in information theory.

Finally, and most importantly, I would like to express my hearty gratitude to my parents Chunying Li and Hao Yang. Their endless love, unfailing support, and continuous encouragement throughout my years of study and research have been the emotional anchor of my life, without which I could not have overcome the ups and downs in the past ten years.

I also recognized that this work would not have been possible without the financial support from UCLA Dissertation Year Fellowship, the National Science Foundation (NSF) under the grants CCF-BSF 1718389, CCF 1717602, CCF-FET 2008728 and CCF 2106213, and in part by AFOSR under the grants FA 9550-20-1-0266 and FA 9550-17-1-0291.

Vita

2016	Bachelor in Electrical Engineering Tsinghua University
2018	Master of Science in Electrical and Computer Engineering University of California, Los Angeles
2016-2021	Research Assistant Electrical and Computer Engineering Department University of California, Los Angeles
2019	System-on-Chip Engineer (Intern) Non-Volatile Memory Solutions Group, Intel Cooperation
2020-2021	Dissertation Year Fellow University of California, Los Angeles

Selected Publications

S. Yang, A. Hareedy, R. Calderbank, and L. Dolecek, "Hierarchical Coding to Enable Scalability and Flexibility in Heterogeneous Cloud Storage," in *Proc. IEEE Global Commun. Conf. (GlobeCOM)*, Hawaii, CA, USA, Dec. 2019, pp. 1–6.

S. Yang, A. Hareedy, R. Calderbank, and L. Dolecek, "Topology-Aware Cooperative Data Protection in Blockchain-Based Decentralized Storage Networks," in *Proc. IEEE International Symposium on Information Theory (ISIT)*, Los Angeles, CA, USA, Jun. 2020, pp. 622–627.

S. Yang, A. Hareedy, S. Venkatasubramanian, R. Calderbank, and L. Dolecek, "GRADE-AO: Towards Near-Optimal Spatially-Coupled Codes With High Memories," in *Proc. IEEE International Symposium on Information Theory (ISIT)*, Melbourne, Victoria, Australia,

July. 2021, pp. 587–592.

S. Yang, C. Schoeny, and L. Dolecek, "Order-Optimal Permutation Codes in the Generalized Cayley Metrics," in *Proc. IEEE Information Theory Workshop (ITW)*, Kaohsiung, Taiwan, China, Nov. 2017, pp. 234–238.

S. Yang, C. Schoeny, and L. Dolecek, "Theoretical Bounds and Constructions of Codes in the Generalized Cayley Metric," on *IEEE Transactions on Information Theory*, vol. 65, no. 8, Aug. 2019, pp. 4746–4763.

CHAPTER 1

Introduction

Nowadays, the burgeoning industry of artificial intelligence has been pervasive and consistently bringing out insatiable appetite for data-driven applications in storage and communication systems. From cloud storage to quantum communications, being efficiently scaled up to accommodate additional workload while simultaneously maintaining desirable properties like high reliability, low latency, and perfect security has always been a major challenge. Error correction coding (ECC) has been demonstrated to be an indispensable technique in overcoming the aforementioned challenges in various data-intensive applications [1, 2, 3, 4, 5, 6, 7, 8].

Cloud storage, known for delivering on demand data storage services to eliminate buying and managing your own data storage infrastructure, provides agile, durable, and low-cost data access that can be efficiently scaled up with the growth of user communities. Cloud storage providers such as Microsoft Azure and Amazon Web Services have become among the most widely deployed public cloud services. Erasure coding has been widely deployed in cloud storage to combat the possible errors incurred from data loss. In particular, files are split into various pieces and distributively stored among nodes in the cloud with added redundancy as a backup for retrieving the data with the presence of component failures. However, while ECC brings up desirable reliability, it inevitably increases the latency for introducing extra reading time on decoding the encoded data. The latency-reliability dilemma is solved by multi-level accessible codes [9, 10, 11, 12, 13, 14, 15, 16], enabling each node to access different sets of helper nodes to retrieve the data, where the sizes of the sets get reduced if the number

of erasures to be recovered is small enough. This architecture is referred to as codes with *hierarchical localities*.

While hierarchical codes efficiently achieve reliability-latency trade-off, other practical challenges emerged from the dynamic nature of the networks are overlooked, which hinder the existing work from being widely deployed to large-scale dynamic networks [17]. Practical management of personal devices faces numerous challenges from component failures, high churn rates, heterogeneous bandwidths and link speeds, in addition to dynamic node balancing for content delivery of hot files [18]. In addition, the recent rise of blockchain technology has initiated a trend of decentralization, accelerating the shift from centralized cloud storage to decentralized storage networks (DSNs), to reach better security at lower costs [19, 20, 21]. In this dissertation, we focus on three major properties, scalability, flexibility, and heterogeneity in coding solutions for both centralized and decentralized cloud storage.

Scalability enables expanding the backbone network to accommodate additional workload, i.e., additional clouds, without rebuilding the entire infrastructure. *Heterogeneity* refers to the property of allowing nonidentical local data lengths and providing unequal local protection, which is important for cloud storage with heterogeneous structures. A heterogeneous structure arises in networks consisting of geographically separated components, and they often store data from different sources. *Flexibility* has been firstly investigated for dynamic data storage systems in [22], and it refers to the property that the local cloud can be split into two smaller local clouds without worsening the global erasure-correction capability nor changing the remaining components. This splitting, for example, is applied when cold data stored at a local cloud become hot unexpectedly. In addition, under the context of DSN, we seek for solutions that also fit into any topology (a property referred to as *topology-adaptivity* later on) with customizable data lengths and redundancies are desired to exploit the existing resources [23, 24, 25, 26, 27]. Topology-adaptivity has been either overlooked or tackled under oversimplified models in existing coding schemes proposed for DSNs [28, 23, 29, 30, 31, 32]. Our proposed hierarchical coding scheme [33, 34, 35] is the first solution to possess all the

aforementioned properties that are desirable in practical cloud storage, while operating on Galois field grows linearly with the maximum local block size.

Challenges from scaling in response to the explosion of data are by no means unique in cloud storage, advanced memory devices also have been consistently paving their way to denser solutions that accommodate the rapid growth of data. From single-level cell (SLC) to more advanced triple-level cell (TLC), and to the most recent 3DXpoint solutions [36], devices with larger storage capacity are always achieved at a cost of higher raw bit error rate (RBER), calling for more advanced and powerful coding solutions to maintain high reliability. Low density parity check (LDPC) codes operating at extremely low frame error rate (FER) region have been the major codes implemented in memory devices and has been intensely studied. Spatially-coupled (SC) LDPC codes, known for their threshold saturation phenomenon, are ideal for data storage systems. SC codes are constructed by partitioning an underlying block code, followed by rearranging and concatenating the partitioned components in a convolutional manner [37, 38, 39, 3, 4]. The number of partitioned components determines the memory of SC codes.

To construct high-performance SC codes, high memories are always desired. Nonetheless, finite-length optimization over SC codes with high memories to expurgate detrimental objects that dominate the error profile is known to be computationally challenging [40, 41, 42, 43, 44]. Heuristic optimization methods strategically avoids the high complexity in combinatorial optimization that aims at globally minimizing the number of detrimental objects [45, 46, 47, 44]. However, high-memory codes designed by purely heuristic methods are unable to reach the potential performance gain that can be achieved through high memories due to lack of theoretical guidance; several of these codes can even be beat by optimally designed SC codes with lower memories under the same constraint length [47]. In this thesis, we propose a novel coding framework, referred to as *gradient descent distributor, algorithmic optimizer (GRADE-AO)*, that unifies seemingly disparate probabilistic and combinatorial approaches, and benefits from both. Simulation results show that codes obtained through our proposed

method notably outperform state-of-the-art codes in a variety of practical settings, including flash memories and hard disk drives.

GRADE-AO is a general framework that applies to a potentially large class of applications; one out of them is information reconciliation in quantum key distribution (QKD). Unconditional security in data communication is currently being deployed in commercial applications. Nonetheless, it faces a number of important challenges including secret key rate, distance, size, cost and practical security. Quantum communications promise to deliver unprecedented levels of security, and to fundamentally change how we transmit and process sensitive data. An important phase of quantum communications is the QKD, which aims at generating private keys used in one-time pad encrypted communications [48, 49, 50, 51, 52, 53]. Error-correction codes, especially LDPC codes, have been demonstrated to be a major technique that enables QKD with desirable secure key rate [5, 54, 55, 56]. High-dimensional quantum key distribution (QKD) protocols based on energy-time entangled photons have been actively investigated for their potential in achieving high secure key rate (SKR). To reach a high SKR, one major challenge is to simultaneously achieve desirable raw key rate (RKR) and photon information efficiency (PIE). Increasing the RKR inevitably leads to degraded channels with errors that are more uniform, thus resulting in lower PIE. This situation calls for robust error correcting codes capable of combating uniform errors. Compared with broadly considered multi-level-coding (MLC) schemes, non-binary (NB) codes are not reliant on the dependencies between bit layers and are more appropriate for high-dimensional QKD. In this work, we develop NB spatially-coupled (SC) irregular repeat accumulate (IRA) codes that strategically combine high performance SC codes and IRA codes. Using the normalized photon efficiency (RPE) as a useful proxy of SKR, where the gain in SKR is lower bounded by the gain in RPE, simulation results show a gain of up to 20.62% of SC-IRA codes in RPE over the MLC scheme, resulting from the improved PIE in the high RKR region.

1.1 Outline of Contributions

In this section, we provide a summary of the contributions in each of the remaining chapters of the dissertation. Our related publications and manuscripts in the references are [33] for Chapter 2, [34] and [35] for Chapter 3, [57] and [58] for Chapter 4, in addition to the relevant publication [59]. Further details about the work contained in this dissertation can be found in the aforementioned publications.

Contributions in Chapter 2

In this chapter, we focus on codes with hierarchical locality and additional properties motivated by their practical importance, including scalability, flexibility, and heterogeneity, as mentioned in previous discussions. We propose a new class of codes based on Cauchy matrices, on a finite field with size that grows linearly with the maximum local codelength. We show that our constructions are able to be scaled up without entirely reinventing the infrastructure, and are adaptive to the dynamic change in usage rates from different local clouds, which are desirable and important in dynamic cloud storage.

Contributions in Chapter 3

In this chapter, we propose a topology-adaptive cooperative data protection scheme for DSNs, which significantly extends our previous work on hierarchical coding for centralized distributed storage. We discuss the recoverable erasure patterns of our proposed scheme, demonstrating that our scheme corrects patterns pertaining to dynamic DSNs. Our scheme achieves faster recovery speed compared with existing network coding methods, and enables an intrinsic information flow from nodes with higher reliability to nodes with lower reliability that are close to them on the network. Our constructions are also proved to be scalable and flexible, making them a construction with great potential to be employed in dynamic DSNs. Most importantly, our coding scheme adapts to DSNs with arbitrary topologies, a property that is especially important in DSNs but has been overlooked in existing works.

Contributions in Chapter 4

In this chapter, we propose the so-called GRADE-AO method, a probabilistic framework that efficiently searches for locally optimal QC-SC codes with arbitrary memories. We obtain a locally optimal edge distribution that minimizes the expected number of the most detrimental objects via gradient descent. Starting from a random partitioning matrix with the derived edge distribution, we then apply a semi-greedy algorithm to find a locally optimal partitioning matrix near it. While the application of GRADE-AO in optimizing the number of short cycles has shown noticeable gains, we focus in this chapter on minimizing the number of more detrimental objects, the concatenated cycles. This finer-grained optimization avoids unnecessary attention on individual cycles which are typically not problematic on their own, especially in codes with high variable node (VN) degrees and irregular codes. Simulation results show that our proposed constructions have a significant performance gain over state-of-the-art codes; this gain is shown to be universal in both waterfall and error floor regions, as well as on channels underlying various practical systems. Future work includes extending the framework to other classes of underlying block codes.

Contributions in Chapter 5

In this chapter, we further extend our GRADE-AO framework to adapt to a larger variety of underlying block codes, especially those perform well in low rate region such that the resultant codes are applicable to QKD. In particular, we generalize GRADE-AO in cycle optimization to adapt to underlying block codes with irregular degrees. Built upon this updated framework and in combination with the well-known IRA codes, we manage to develop a finite-length SC-IRA code optimization that efficiently optimizes over the number of cycles in the SC codes with an underlying IRA code. Simulation results demonstrate a nontrivial improvement of our proposed scheme over the MLC/MSD scheme in combating global errors and channel variations in QKD.

CHAPTER 2

Hierarchical Coding for Centralized Cloud Storage

2.1 Introduction

Codes offering hierarchical locality have been intensely studied because of their ability to reduce the average reading time in various erasure-resilient data storage applications including Flash storage, redundant array of independent disks (RAID) storage, cloud storage, etc. [11]. Codes with shorter block lengths offer lower latency, but they provide limited erasure-correction capability in a cloud storage system. To deal with more erasures, longer codes can be employed. However, since a simultaneous occurrence of a large number of erasures is a rare event, longer codes result in unnecessary extra reading cost, and are on average inefficient. Therefore, maintaining low latency while simultaneously recovering from a potentially large number of erasures is one of the major challenges in cloud storage. Codes with hierarchical locality have been shown to address this issue by providing multi-level access in cloud storage, which enables the data to be read through a chain of network components with increasing data lengths from top to bottom; this architecture is exploited to increase the overall erasure-correction capability[9].

In the literature, codes offering double-level access have been intensely studied[11, 9, 15, 14, 12, 22]; these codes are applicable in double-level cloud storage. In this configuration, p consecutive local messages are jointly encoded into p correlated local codewords. Each local codeword is stored at the neighboring servers of the corresponding local cloud. The codes are

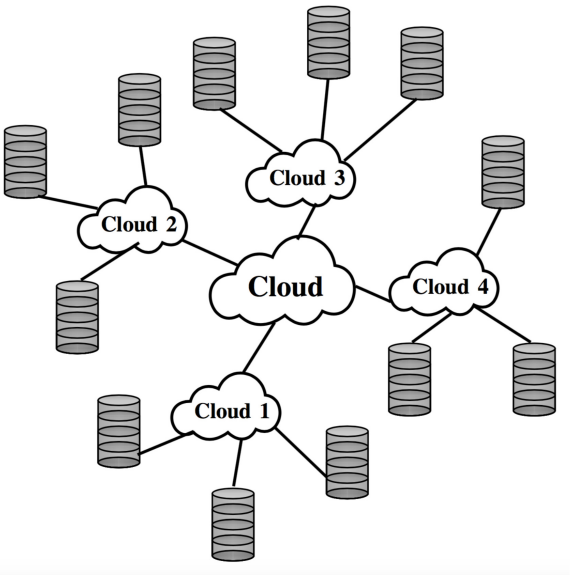


Figure 2.1: Double-level cloud storage. Servers connected to local clouds store the local codewords; the local clouds are connected to a central cloud.

designed such that each local message can be successfully decoded from the corresponding local codeword when there are fewer than d_1 local erasures, and the global codeword provides extra protection against $(d_2 - d_1)$ unexpected errors in a local codeword, for some $d_2 > d_1$. An example having $p = 4$ is in Fig. 1. Suppose $d_1 = 2$ and $d_2 = 3$. When there is at most 1 server failure, accessing the servers connected to cloud 1 is sufficient to successfully decode the data stored in cloud 1. If the number of server failures in cloud 1 is 2, the data can still be obtained through accessing all the servers. Codes with hierarchical locality are a generalized extension of double-level accessible codes, in which more than two levels of access are allowed and are naturally suitable for cloud storage with multiple layers. An application in which these codes are needed is hybrid cloud storage [60].

Along with *hierarchical locality* discussed previously, it is also important for the coding schemes to support scalable, heterogeneous, and flexible cloud storage[17]. *Scalability* enables expanding the backbone network to accommodate additional workload, i.e., additional clouds, without rebuilding the entire infrastructure. *Heterogeneity* refers to the property of allowing nonidentical local data lengths and providing unequal local protection, which

is important for cloud storage with heterogeneous structures. A heterogeneous structure arises in networks consisting of geographically separated components, and they often store data from different sources. *Flexibility* has been firstly investigated for dynamic data storage systems in [22], and it refers to the property that the local cloud can be split into two smaller local clouds without worsening the global erasure-correction capability nor changing the remaining components. This splitting, for example, is applied when cold data stored at a local cloud become hot unexpectedly.

Various codes offering hierarchical locality have been studied. Cassuto *et al.*[11] presented so-called multi-block interleaved codes that provide double-level access; this work introduced the concept of multi-level access. The family of integrated-interleaved (I-I) codes, including generalized integrated interleaved (GII) codes and extended integrated interleaved (EII) codes, has been a major prototype for codes with multi-level access [9, 12, 14, 15]. GII codes have the advantage of correcting a large set of error patterns, but the distribution of the data symbols is highly restricted, and all the local codewords are equally protected. EII codes are extensions of GII codes with double-level access, where specific arrangements of data symbols have been investigated, mitigating the aforementioned restriction. However, no similar study has been proposed for GII codes with hierarchical locality. Therefore, I-I codes are more suitable for applications where heterogeneity and flexibility are less important. Sum-rank codes are another family of codes that is proposed for dynamic distributed storage offering double-level access[22]. These codes are maximally recoverable, flexible, and allow unequal protection for local data. However, sum-rank codes require a finite field size that grows exponentially with the maximum local block length, which is a major obstacle to being implemented in real world applications.

In this chapter, we introduce code constructions with hierarchical locality and a small field size that achieve scalability, heterogeneity, and flexibility. The chapter is organized as follows. In Section 2.2, we introduce the notation and preliminaries. In Section 2.3, we present a new construction of codes offering hierarchical locality that is based on Cauchy

Reed Solomon (CRS) codes. This construction requires a field size that grows linearly with the maximum local codelength. In Section 2.4, we then show that our coding scheme is scalable, heterogeneous, and flexible. Finally, we summarize our results in Section 4.7.

2.2 Notation and Preliminaries

Throughout the rest of this chapter, $[N]$ refers to $\{1, 2, \ldots, N\}$, and $[a : b]$ refers to $\{a, a + 1, \ldots, b\}$. Denote the all zero vector of length s by $\mathbf{0}_s$. Similarly, the all zero matrix of size $s \times t$ is denoted by $\mathbf{0}_{s\times t}$. The alphabet field, denoted by $\mathrm{GF}(q)$, is a Galois field of size q, where q is a power of a prime. For a vector $\mathbf{v}$ of length n, v_i, $1 \leq i \leq n$, represents the i-th component of $\mathbf{v}$, and $\mathbf{v}[a : b] = (v_a, \ldots, v_b)$. For a matrix $\mathbf{M}$ of size $a \times b$, $\mathbf{M}[i_1 : i_2, j_1 : j_2]$ represents the sub-matrix $\mathbf{M}'$ of $\mathbf{M}$ such that $(\mathbf{M}')_{i-i_1+1,j-j_1+1} = (\mathbf{M})_{i,j}$, $i \in [i_1 : i_2]$, $j \in [j_1 : j_2]$. All indices start from 1.

2.2.1 Notation and Definitions

Let $\mathbf{m}$ and $\mathbf{c}$ represent messages and codewords, respectively. A set $\mathcal{C}$ is called an $(n, k, d)_q$-code if $\mathcal{C} \subset \mathrm{GF}(q)^n$, $\dim(\mathcal{C}) = k$, and $\min_{\mathbf{c}_1,\mathbf{c}_2\in\mathcal{C},\mathbf{c}_1\neq\mathbf{c}_2} d_{\mathrm{H}}(\mathbf{c}_1, \mathbf{c}_2) = d$, where d_{H} refers to the Hamming distance. We next define a family of codes with double-level access. Note that our discussion is restricted to linear block codes.

Definition 1. *Let $p, q \in \mathbb{N}$. Let $\mathbf{n} = (n_1, n_2, \ldots, n_p) \in \mathbb{N}^p$, $\mathbf{k} = (k_1, k_2, \ldots, k_p) \in \mathbb{N}^p$, $\mathbf{D} \in \mathbb{N}^{2\times p}$, $(\mathbf{D})_{x,y} = d_{x,y}$, where $d_{1,x} < d_{2,x}$, $k_x < n_x$, for all $x, y \in [p]$.*

Let $n = n_1 + n_2 + \cdots + n_p$. Let $s_0 = 0$ and $s_x = n_1 + n_2 + \cdots + n_x$, $x \in [p]$. Let $\mathbf{c}_x$ denote $\mathbf{c}[s_{x-1} + 1 : s_x]$ and let $\mathbf{m}_x$ denote the message corresponding to $\mathbf{c}_x$, for $x \in [p]$. A set $\mathcal{C} \subset \mathrm{GF}(q)^n$ is called an $(\mathbf{n}, \mathbf{k}, \mathbf{D}, p)_q$-code if the following conditions are satisfied:

1. *Let $\mathcal{C}_x = \{\mathbf{c}[s_{x-1} + 1 : s_x] : \mathbf{c} \in \mathcal{C}\}$, $x \in [p]$. Each $\mathcal{C}_x$ is an $(n_x, k_x, d_{1,x})_q$-code.*
2. *Let $\mathcal{A}_x = \{\mathbf{c}[s_{x-1} + 1 : s_x] : \mathbf{c} \in \mathcal{C}, \mathbf{c}[s_{y-1} + 1 : s_y] = \mathbf{0}_{n_y}, \forall y \in [p] \setminus \{x\}\}$, $x \in [p]$. Each $\mathcal{A}_x$ is an $(n_x, k_x, d_{2,x})_q$-code.*

Any $(\mathbf{n}, \mathbf{k}, \mathbf{D}, p)_q$-code specified according to Definition 1 corrects $(d_{1,x} - 1)$ erasures in the i-th local codeword via local access, and corrects additional $(d_{2,x} - d_{1,x})$ erasures through global access when other local codewords are all correctable via local access. Following this notation, Definition 2 extends Definition 1 into the triple-level case.

Definition 2. *Let* $q, p_0 \in \mathbb{N}$, $\mathbf{p} = (p_1, p_2, \ldots, p_{p_0}) \in \mathbb{N}^{p_0}$, $p = p_1 + p_2 + \cdots + p_{p_0}$. *Let* $\mathbf{n} = (\mathbf{n}_1, \mathbf{n}_2, \ldots, \mathbf{n}_{p_0}) \in \mathbb{N}^{p_0}$, $\mathbf{k} = (\mathbf{k}_1, \mathbf{k}_2, \ldots, \mathbf{k}_{p_0}) \in \mathbb{N}^{p_0}$, *where* $\mathbf{n}_x = (n_{x,1}, n_{x,2}, \ldots, n_{x,p_x}) \in \mathbb{N}^{p_x}$, $\mathbf{k}_x = (k_{x,1}, k_{x,2}, \ldots, k_{x,p_x}) \in \mathbb{N}^{p_x}$, *for all* $x \in [p_0]$.

Let $t_0 = 0$, $t_x = p_1 + p_2 + \cdots + p_x$, $x \in [p_0]$. *Suppose* $\mathbf{D} \in \mathbb{N}^{3 \times p}$. *Let* $d_{l,x,i} = (\mathbf{D})_{l,t_{x-1}+i}$, $l \in [3]$ *so that* $d_{1,x,i} < d_{2,x,i} < d_{3,x,i}$, *for* $x \in [p_0]$ *and* $i \in [p_x]$. *Let* $\mathbf{D}_x = \mathbf{D}[1:2, t_{x-1}+1 : t_x]$, $x \in [p_0]$. *Let* $n_x = n_{x,1} + n_{x,2} + \cdots + n_{x,p_x}$ *for all* $x \in [p_0]$. *Let* $n = n_1 + n_2 + \cdots + n_{p_0}$. *Let* $s_0 = 0$, $s_x = n_1 + n_2 + \cdots + n_x$, $x \in [p_0]$. *Let* $s_{x,0} = s_x$, $s_{x,i} = s_x + n_{x,1} + n_{x,2} + \cdots + n_{x,i}$, *for all* $x \in [p_0]$ *and* $i \in [p_x]$. *Let* $\mathbf{c}_{x,i}$ *denote* $\mathbf{c}[s_{x,i-1}+1 : s_{x,i}]$ *and let* $\mathbf{m}_{x,i}$ *denote the message corresponding to* $\mathbf{c}_{x,i}$, *for* $x \in [p_0]$, $i \in [p_x]$. *A set* $\mathcal{C} \subset \mathrm{GF}(q)^n$ *is called an* $(\mathbf{n}, \mathbf{k}, \mathbf{D}, p_0, \mathbf{p})_q$*-code if the following conditions are satisfied:*

1. *Let* $\mathcal{C}_x = \{\mathbf{c}[s_{x-1}+1 : s_x] : \mathbf{c} \in \mathcal{C}\}$, $x \in [p_0]$. *Each* $\mathcal{C}_x$ *is an* $(\mathbf{n}_x, \mathbf{k}_x, \mathbf{D}_x, p_x)_q$*-code.*

2. *Let* $\mathcal{A}_{x,i} = \{\mathbf{c}[s_{x,i-1}+1 : s_{x,i}] : \mathbf{c} \in \mathcal{C}, \mathbf{c}[s_{y,j-1}+1 : s_{y,j}] = \mathbf{0}_{n_{y,j}}, \forall y \in [p_0], j \in [p_y], (x,i) \neq (y,j)\}$. *Each* $\mathcal{A}_x$ *is an* $(n_{x,i}, k_{x,i}, d_{3,x,i})_q$*-code.*

This definition can be easily generalized into codes with more than three levels of access. For simplicity, we constrain our discussion to the triple-level case.

2.2.2 Cauchy Matrices

Cauchy matrices are the key component in the construction that we will introduce shortly.

Before we describe the constructions in detail, we first introduce the so-called **Cauchy matrices** that are used as major components in the generator matrices of our codes. Codes based on Cauchy matrices, the so-called Cauchy Reed-Solomon (CRS) codes, have been

studied in [61, 62]. CRS codes present desirable properties, as discussed later, and have been proposed to be applied to distributed storage systems in [63, 64]. In our work, we further exploit the scaling property of CRS codes, which makes them an ideal choice to accommodate hierarchical access on arbitrarily deployed nodes in DSNs.

Definition 3. (Cauchy matrix) *Let $s, t \in \mathbb{N}$ and $\mathrm{GF}(q)$ be a Galois field of size q. Suppose $a_1, \ldots, a_s, b_1, \ldots, b_t$ are $s+t$ distinct elements in $\mathrm{GF}(q)$. The following matrix is known as a* ***Cauchy matrix****,*

$$\begin{bmatrix} \frac{1}{a_1-b_1} & \frac{1}{a_1-b_2} & \cdots & \frac{1}{a_1-b_t} \\ \frac{1}{a_2-b_1} & \frac{1}{a_2-b_2} & \cdots & \frac{1}{a_2-b_t} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{a_s-b_1} & \frac{1}{a_s-b_2} & \cdots & \frac{1}{a_s-b_t} \end{bmatrix}$$

We denote this matrix by $\mathbf{Y}(a_1, \ldots, a_s; b_1, \ldots, b_t)$, and refer to sequences $(a_1, a_2, \ldots, a_s)$, $(b_1, b_2, \ldots, b_t)$ as the ***row indicator*** *and the* ***column indicator*** *of the specified Cauchy matrix, respectively.*

Cauchy matrices possess desirable properties that make them an ideal alternative to Vandermonde matrices, the major components of the parity-check matrices of Reed-Solomon (RS) codes, as the parity-computing (non-systematic) components in systematic generator matrices of some maximum distance separable (MDS) codes with low encoding and decoding complexities [63]. Cauchy matrices are **totally invertible**, i.e., every square sub-matrix of a Cauchy matrix is invertible. Therefore, horizontally concatenating a Cauchy matrix with another Cauchy matrix having an identical row indicator but a non-overlapping column indicator results in a third Cauchy matrix. Similarly, vertically concatenating a Cauchy matrix with another Cauchy matrix having an identical column indicator but a non-overlapping row indicator also results in a third Cauchy matrix. This property, referred to as the scaling property previously, is desirable for hierarchical access in topology-adaptive DSNs. Moreover, Lemma 1 presents another useful property about Cauchy matrices, which will be used repeatedly in this chapter.

Cauchy matrices are *totally invertible*, i.e., every square sub-matrix of a Cauchy matrix is invertible. The inverse of a given Cauchy matrix can be explicitly computed using algorithms of lower complexity than those for inverting Vandermonde matrices. Lemma 1 presents a useful result about Cauchy matrices that will be used repeatedly in this chapter.

Lemma 1. *Let $s, t, r \in \mathbb{N}$ such that $t - s < r \leq t$, and $\mathbf{A} \in \mathrm{GF}(q)^{s \times t}$. If $\mathbf{A}$ is a Cauchy matrix, then the following matrix $\mathbf{M}$ is a parity-check matrix of an $(s+r, s+r-t, t+1)_q$-code*[1]*,*

$$\mathbf{M} = \begin{bmatrix} \mathbf{A} \\ -\mathbf{I}_r \ \mathbf{0}_{r \times (t-r)} \end{bmatrix}^{\mathrm{T}}.$$

Proof. The parity-check matrix of an $(s+r, s+r-t, t+1)_q$-code satisfies the property that every t columns of this matrix are linearly independent. Therefore, we only need to prove that every t rows of $\mathbf{M}^{\mathrm{T}}$ are linearly independent. We prove Lemma 1 by contradiction. Suppose there exist t rows from $\mathbf{M}^{\mathrm{T}}$ that are linearly dependent. Suppose a of these linearly dependent rows $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_a$ are from $\mathbf{A}$, and the other $t-a$ rows $\mathbf{r}_{a+1}, \mathbf{r}_{a+2}, \ldots, \mathbf{r}_t$ are from $\left[-\mathbf{I}_r \ \mathbf{0}_{r \times (t-r)}\right]$, where $0 \leq t - a \leq r$. Suppose the entries with -1 in $\mathbf{r}_{a+1}, \mathbf{r}_{a+2}, \ldots, \mathbf{r}_t$ are located in the $i_1, i_2 \ldots, i_{t-a}$-th columns of $\mathbf{M}^{\mathrm{T}}$, then $i_p \leq r$ for all $1 \leq p \leq t-a$. Observe that $[t]$ is the set of indices of all columns in $\mathbf{M}^{\mathrm{T}}$. Suppose $[t] \setminus \{i_1, i_2, \ldots, i_{t-a}\} = \{j_1, j_2, \ldots, j_a\}$. Then, the $a \times a$ sub-matrix of the intersection of the rows $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_a$ and the $j_1, j_2, \ldots, j_a$-th columns of $\mathbf{A}$ is singular. A contradiction. □

2.3 Codes for Multi-Level Access

Following the definitions and notation introduced in Section 2.2, we present a CRS-based code with double-level access in Section 2.3.1. Then, we extend our construction into a triple-level case in Section 2.3.2.

[1]Note that when q is a power of 2, the minus operand can be removed, as shown in Example 1, since subtraction and addition are equivalent on the Galois field $\mathrm{GF}(q)$ in this case.

2.3.1 Codes with Double-Level Access

In this subsection, we provide a construction of codes offering double-level access based on the CRS codes. Note that the generator matrix of any systematic code with double-level access has the following structure:

$$\mathbf{G} = \left[\begin{array}{c|c|c|c|c|c|c} \mathbf{I}_{k_1} & \mathbf{A}_{1,1} & \mathbf{0} & \mathbf{A}_{1,2} & \dots & \mathbf{0} & \mathbf{A}_{1,p} \\ \hline \mathbf{0} & \mathbf{A}_{2,1} & \mathbf{I}_{k_2} & \mathbf{A}_{2,2} & \dots & \mathbf{0} & \mathbf{A}_{2,p} \\ \hline \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \hline \mathbf{0} & \mathbf{A}_{p,1} & \mathbf{0} & \mathbf{A}_{p,2} & \dots & \mathbf{I}_{k_p} & \mathbf{A}_{p,p} \end{array}\right]. \tag{2.1}$$

Construction 1. *(CRS-based code) Let $p \in \mathbb{N}$, $k_1, k_2, \dots, k_p \in \mathbb{N}$, $n_1, n_2, \dots, n_p \in \mathbb{N}$, $\delta_1, \delta_2, \dots, \delta_p \in \mathbb{N}$ and $\delta = \delta_1 + \delta_2 + \cdots + \delta_p$, with $r_x = n_x - k_x > 0$ for all $x \in [p]$. Let $GF(q)$ be a finite field such that $q \geq \max_{x\in[p]}\{n_x\} + \delta$.*

For each $x \in [p]$, let $a_{x,i}$, $b_{x,j}$, $i \in [k_x + \delta_x]$, $j \in [r_x - \delta_x + \delta]$, be distinct elements of $\mathrm{GF}(q)$. *Consider the Cauchy matrix $\mathbf{T}_x \in \mathrm{GF}(q)^{(k_x+\delta_x)\times(r_x-\delta_x+\delta)}$ such that $\mathbf{T}_x = \mathbf{Y}(a_{x,1}, \dots, a_{x,k_x+\delta_x}; b_{x,1}, \dots, b_{x,r_x-\delta_x+\delta})$. For each $x \in [p]$, we obtain $\{\mathbf{B}_{x,i}\}_{i\in[p]\setminus\{x\}}$, $\mathbf{U}_x$, $\mathbf{A}_{x,x}$, according to the following partition of $\mathbf{T}_x$,*

$$\mathbf{T}_x = \left[\begin{array}{c|c|c|c} \mathbf{A}_{x,x} & \mathbf{B}_{x,1} & \dots & \mathbf{B}_{x,p} \\ \hline \mathbf{U}_x & \multicolumn{3}{c}{\mathbf{Z}_x} \end{array}\right], \tag{2.2}$$

where $\mathbf{A}_{x,x} \in \mathrm{GF}(q)^{k_x\times r_x}$, $\mathbf{B}_{x,i} \in \mathrm{GF}(q)^{k_x\times\delta_i}$, $\mathbf{U}_x \in \mathrm{GF}(q)^{\delta_x\times r_x}$. Moreover, $\mathbf{A}_{x,y} = \mathbf{B}_{x,y}\mathbf{U}_y$, for $x \neq y$.

Matrices $\mathbf{A}_{x,x}$ and $\mathbf{A}_{x,y}$ are substituted in $\mathbf{G}$ specified in (3.1), for all $x, y \in [p]$, $x \neq y$. Let $\mathcal{C}_1$ represent the code with generator matrix $\mathbf{G}$.

Lemma 2. *Following the notation in Definition 1, let $d_{1,x} = r_x - \delta_x + 1$, $d_{2,x} = r_x - \delta_x + \delta + 1$, for $x \in [p]$. Then, code $\mathcal{C}_1$ specified in Construction 3 is an $(\mathbf{n}, \mathbf{k}, \mathbf{D}, p)_q$-code.*

Sketch of the proof. For each $x \in [p]$, define $\mathbf{y}_x = \sum_{y\in[p],y\neq x} \mathbf{m}_y\mathbf{B}_{y,x}$. It follows from $\mathbf{mG} = \mathbf{c}$

Table 2.1: Polynomial and normal forms of GF(2^4)

0	0000	β^4	1100	β^8	1010	β^{12}	1111
β	0100	β^5	0110	β^9	0101	β^{13}	1011
β^2	0010	β^6	0011	β^{10}	1110	β^{14}	1001
β^3	0001	β^7	1101	β^{11}	0111	$\beta^{15}=1$	1000

and (3.1) that for $x \in [p]$, $\mathbf{c}_x = [\mathbf{m}_x, \mathbf{m}_x\mathbf{A}_{x,x} + \mathbf{y}_x\mathbf{U}_x]$. Define the local parity-check matrix $\mathbf{H}_x^{\mathrm{L}}$ and the global parity-check matrix $\mathbf{H}_x^{\mathrm{G}}$, for each $x \in [p]$, as follows:

$$\mathbf{H}_x^{\mathrm{G}} = \left[\begin{array}{c|c|c|c} \mathbf{A}_{x,x} & \mathbf{B}_{x,1} & \cdots & \mathbf{B}_{x,p} \\ \hline -\mathbf{I}_{r_x} & \multicolumn{3}{c}{\mathbf{0}_{r_x \times \delta - \delta_x}} \end{array}\right]^{\mathrm{T}}, \mathbf{H}_x^{\mathrm{L}} = \left[\begin{array}{c} \mathbf{A}_{x,x} \\ -\mathbf{I}_{r_x} \\ \mathbf{U}_x \end{array}\right]^{\mathrm{T}}.$$

We next prove the equations of the local distance $d_{1,x} = r_x - \delta_x + 1$ and the global distance $d_{2,x} = r_x - \delta_x + \delta + 1$ using $\mathbf{H}_x^{\mathrm{L}}$ and $\mathbf{H}_x^{\mathrm{G}}$, $x \in [p]$.

To prove the equation of the local distance, let $\tilde{\mathbf{c}}_x = [\mathbf{c}_x, \mathbf{y}_x]$. Then, one can show that $\tilde{\mathbf{c}}_x$ belongs to a code $\mathcal{C}_x^{\mathrm{L}}$ with the local parity-check matrix $\mathbf{H}_x^{\mathrm{L}}$. From Lemma 1, $\mathcal{C}_x^{\mathrm{L}}$ is an $(n_x + \delta_x, k_x, r_x + 1)_q$-code. Therefore, any r_x erasures in $\tilde{\mathbf{c}}_x$ are correctable. Provided that $\mathbf{y}_x$ has length δ_x, we can consider the entries of $\mathbf{y}_x$ as erasures and thus any $(r_x - \delta_x)$ erasures in the remaining part of $\tilde{\mathbf{c}}_x$, i.e., $\mathbf{c}_x$, can be corrected. Therefore, $d_{1,x} = r_x - \delta_x + 1$.

To prove the equation of the global distance, assume all the local codewords except for $\mathbf{c}_x$ are successfully decodable locally. Then, for each $x \in [p]$, $\mathbf{y}_x$ and $\mathbf{s}_x = [\mathbf{m}_x\mathbf{B}_{x,1}, \ldots, \mathbf{m}_x\mathbf{B}_{x,p}]$ are computable. Let $\bar{\mathbf{c}}_x = \mathbf{c}_x - [\mathbf{0}_{k_x}, \mathbf{y}_x\mathbf{U}_x]$, then one can show that $\mathbf{H}_x^{\mathrm{G}}\bar{\mathbf{c}}_x^{\mathrm{T}} = [\mathbf{0}_{r_x}, \mathbf{s}_x]^{\mathrm{T}}$. From Lemma 1 and from the construction of $\mathbf{H}_x^{\mathrm{G}}$, any $(r_x - \delta_x + \delta)$ erasures in $\bar{\mathbf{c}}_x$ are correctable, thus $(r_x - \delta_x + \delta)$ erasures in $\mathbf{c}_x$ are also correctable. Therefore, $d_{2,x} = r_x - \delta_x + \delta + 1$. □

We next provide a working example for codes in Construction 3. For simplicity, we let all the local codeword lengths and local data lengths be equal. However, the construction itself allows them to be unequal.

$$\mathbf{T}_1 = \mathbf{T}_2 = \left[\begin{array}{c|c} \mathbf{A}_{1,1} & \mathbf{B}_{1,2} \\ \hline \mathbf{U}_1 & \mathbf{Z}_1 \end{array}\right] = \left[\begin{array}{c|c} \mathbf{A}_{2,2} & \mathbf{B}_{2,1} \\ \hline \mathbf{U}_2 & \mathbf{Z}_2 \end{array}\right]$$
$$= \left[\begin{array}{ccc|c} \frac{1}{\beta-\beta^8} & \frac{1}{\beta-\beta^9} & \frac{1}{\beta-\beta^{10}} & \frac{1}{\beta-\beta^{11}} \\ \frac{1}{\beta^2-\beta^8} & \frac{1}{\beta^2-\beta^9} & \frac{1}{\beta^2-\beta^{10}} & \frac{1}{\beta^2-\beta^{11}} \\ \frac{1}{\beta^3-\beta^8} & \frac{1}{\beta^3-\beta^9} & \frac{1}{\beta^3-\beta^{10}} & \frac{1}{\beta^3-\beta^{11}} \\ \hline \frac{1}{\beta^7-\beta^8} & \frac{1}{\beta^7-\beta^9} & \frac{1}{\beta^7-\beta^{10}} & \frac{1}{\beta^7-\beta^{11}} \end{array}\right] = \left[\begin{array}{ccc|c} \beta^5 & \beta^{12} & \beta^7 & \beta^9 \\ 1 & \beta^4 & \beta^{11} & \beta^6 \\ \beta^2 & \beta^{14} & \beta^3 & \beta^{10} \\ \hline \beta^4 & 1 & \beta^9 & \beta^7 \end{array}\right]. \tag{2.3}$$

Example 1. *Let* $q = 2^4$, $p = 2$, $r = r_1 = r_2 = 3$, $\delta' = \delta_1 = \delta_2 = 1$, $k = k_1 = k_2 = 3$, $n = n_1 = n_2 = k + r = 6$, $\delta = \delta_1 + \delta_2 = 2$. *Then,* $d_1 = r - \delta' + 1 = 3 - 1 + 1 = 3$, $d_2 = r - \delta' + \delta + 1 = 3 - 1 + 2 + 1 = 5$. *Choose a primitive polynomial over* GF(2)*:* $g(X) = X^4 + X + 1$. *Let* β *be a root of* $g(X)$*, then* β *is a primitive element of* $\mathrm{GF}(2^4)$. *The binary representation of all the symbols in* $\mathrm{GF}(2^4)$ *is specified in Table 3.1.*

Let $\mathbf{A}_{1,1} = \mathbf{A}_{2,2}$, $\mathbf{B}_{1,2} = \mathbf{B}_{2,1}$, $\mathbf{U}_1 = \mathbf{U}_2$, *and* $\mathbf{T}_1 = \mathbf{T}_2$ *as specified in (2.3). Therefore,*

$$\mathbf{A}_{1,2} = \mathbf{A}_{2,1} = \mathbf{B}_{2,1}\mathbf{U}_1 = \begin{bmatrix} \beta^{13} & \beta^9 & \beta^3 \\ \beta^{10} & \beta^6 & 1 \\ \beta^{14} & \beta^{10} & \beta^4 \end{bmatrix}.$$

Then, the generator matrix $\mathbf{G}$ *is specified as follows,*

$$\left[\begin{array}{ccc|ccc|ccc|ccc} 1 & 0 & 0 & \beta^5 & \beta^{12} & \beta^7 & 0 & 0 & 0 & \beta^{13} & \beta^9 & \beta^3 \\ 0 & 1 & 0 & 1 & \beta^4 & \beta^{11} & 0 & 0 & 0 & \beta^{10} & \beta^6 & 1 \\ 0 & 0 & 1 & \beta^2 & \beta^{14} & \beta^3 & 0 & 0 & 0 & \beta^{14} & \beta^{10} & \beta^4 \\ \hline 0 & 0 & 0 & \beta^{13} & \beta^9 & \beta^3 & 1 & 0 & 0 & \beta^5 & \beta^{12} & \beta^7 \\ 0 & 0 & 0 & \beta^{10} & \beta^6 & 1 & 0 & 1 & 0 & 1 & \beta^4 & \beta^{11} \\ 0 & 0 & 0 & \beta^{14} & \beta^{10} & \beta^4 & 0 & 0 & 1 & \beta^2 & \beta^{14} & \beta^3 \end{array}\right].$$

Suppose $\mathbf{m}_1 = (1, \beta, \beta^2)$, $\mathbf{m}_2 = (\beta, 1, 0)$, *then* $\mathbf{c}_1 = (1, \beta, \beta^2, \beta^{14}, 0, 0)$ *and* $\mathbf{c}_2 = (\beta, 1, 0, \beta^6, 0,$

β^{13}). *Moreover,* $\mathbf{H}_1^{\mathrm{L}}$ *and* $\mathbf{H}_1^{\mathrm{G}}$ *are specified as follows,*

$$\mathbf{H}_1^{\mathrm{G}} = \begin{bmatrix} \beta^5 & \beta^{12} & \beta^7 & \beta^9 \\ 1 & \beta^4 & \beta^{11} & \beta^6 \\ \beta^2 & \beta^{14} & \beta^3 & \beta^{10} \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}^{\mathrm{T}}, \mathbf{H}_1^{\mathrm{L}} = \begin{bmatrix} \beta^5 & \beta^{12} & \beta^7 \\ 1 & \beta^4 & \beta^{11} \\ \beta^2 & \beta^{14} & \beta^3 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ \beta^4 & 1 & \beta^9 \end{bmatrix}^{\mathrm{T}}.$$

According to Construction 3, $\mathbf{G}$ *is the generator matrix of a double-level accessible code that corrects* 2 *local erasures by local access and corrects* 2 *extra erasures within a single local cloud by global access. In the following, we denote the erased version of* $\mathbf{c}_1$ *by* $\mathbf{c}_1'$*, and erased symbols by* e_i*,* $i \in \mathbb{N}$*.*

As an example of decoding by local access, suppose $\mathbf{c}_1' = (1, e_1, \beta^2, e_2, 0, 0)$*. Then, the erased elements of* $\tilde{\mathbf{c}}_1 = (1, e_1, \beta^2, e_2, 0, 0, e_3)$ *can be retrieved using* $\mathbf{H}_1^{\mathrm{L}}$ *as the parity-check matrix. In particular, we solve* $\mathbf{H}_1^{\mathrm{L}}\tilde{\mathbf{c}}_1^{\mathrm{T}} = (0,0,0)^{\mathrm{T}}$ *for* e_1, e_2, e_3 *and obtain* $(e_1, e_2, e_3) = (\beta, \beta^{14}, \beta^7)$*. We have decoded* $\mathbf{c}_1$ *successfully.*

As an example of decoding by global access, suppose $\mathbf{c}_1' = (e_1, e_2, \beta^2, e_3, e_4, 0)$*, and* $\mathbf{c}_2$ *has been locally decoded successfully. Then,* $\mathbf{c}_2 = (\beta, 1, 0, \beta^6, 0, \beta^{13})$ *implies that* $\mathbf{m}_1\mathbf{B}_{1,2}\mathbf{U}_2 = (\beta^6, 0, \beta^{13}) - \beta \cdot (\beta^5, \beta^{12}, \beta^7) - 1 \cdot (1, \beta^4, \beta^{11}) = (1, \beta^{11}, \beta^5)$*. Since* $\mathbf{U}_2 = (\beta^4, 1, \beta^9)$*, we obtain* $\mathbf{m}_1\mathbf{B}_{1,2} = \beta^{11}$*. Moreover, we compute* $\mathbf{m}_2\mathbf{B}_{2,1}\mathbf{U}_1 = (\beta^{11}, \beta^7, \beta)$*. Let* $\bar{\mathbf{c}}_1 = \mathbf{c}_1' - (0,0,0,\beta^{11},\beta^7,\beta) = (e_1', e_2', \beta^2, e_3', e_4', \beta)$*. Then, we solve* $\mathbf{H}_1^{\mathrm{G}}\bar{\mathbf{c}}_1^{\mathrm{T}} = (0,0,0,\beta^{11})^{\mathrm{T}}$ *and obtain* $(e_1', e_2', e_3', e_4') = (1, \beta, \beta^{10}, \beta^7)$*. Therefore,* $e_1 = e_1' = 1$*,* $e_2 = e_2' = \beta$*,* $e_3 = e_3' + \beta^{11} = \beta^{14}$*,* $e_4 = e_4' + \beta^7 = 0$*, and we have decoded* $\mathbf{c}_1$ *successfully.*

2.3.2 Codes with Hierarchical Locality

Based on the double-level accessible codes presented in Section 2.3.1, we present a class of codes with hierarchical locality in Construction 2. For simplicity, we just present a construction with triple-level access. Note that the coding scheme itself can be naturally extended to have more than three levels. A detailed explanation of the subscripts used in the following discussion is provided in [33].

A generator matrix of such a code is as follows:

$$\mathbf{G} = \left[\begin{array}{c|c|c|c} \mathbf{F}_{1,1} & \mathbf{F}_{1,2} & \dots & \mathbf{F}_{1,p_0} \\ \hline \mathbf{F}_{2,1} & \mathbf{F}_{2,2} & \dots & \mathbf{F}_{2,p_0} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline \mathbf{F}_{p_0,1} & \mathbf{F}_{p_0,2} & \dots & \mathbf{F}_{p_0,p_0} \end{array}\right], \tag{2.4}$$

where for any $x \in [p_0]$,

$$\mathbf{F}_{x,x} = \left[\begin{array}{c|c|c|c|c} \mathbf{I}_{k_{x,1}} & \mathbf{A}_{x,x;1,1} & \dots & \mathbf{0} & \mathbf{A}_{x,x;1,p_x} \\ \hline \vdots & \ddots & \ddots & \vdots & \vdots \\ \hline \mathbf{0} & \mathbf{A}_{x,x;p_x,1} & \dots & \mathbf{I}_{k_{x,p_x}} & \mathbf{A}_{x,x;p_x,p_x} \end{array}\right], \tag{2.5}$$

is a generator matrix of a code offering double-level access, and

$$\mathbf{F}_{x,y} = \left[\begin{array}{c|c|c|c|c} \mathbf{0} & \mathbf{A}_{x,y;1,1} & \dots & \mathbf{0} & \mathbf{A}_{x,y;1,p_y} \\ \hline \vdots & \ddots & \ddots & \vdots & \vdots \\ \hline \mathbf{0} & \mathbf{A}_{x,y;p_x,1} & \dots & \mathbf{0} & \mathbf{A}_{x,y;p_x,p_y} \end{array}\right]. \tag{2.6}$$

Properties of $\mathbf{F}_{x,x}, \mathbf{F}_{x,y}$ are to be discussed later.

Construction 2. *Let $p_0 \in \mathbb{N}$, $\mathbf{p} = (p_1, \dots, p_{p_0}) \in \mathbb{N}^{p_0}$. Let $k_{x,i}, n_{x,i}, \delta_{x,i}, \gamma_x \in \mathbb{N}$, for $x \in [p_0]$ and $i \in [p_x]$, such that $r_{x,i} = n_{x,i} - k_{x,i} > 0$ and $2\gamma_x < \min_{i\in[p_x]}\{r_{x,i} - \delta_{x,i}\}$. Let $\delta_x = \delta_{x,1} + \cdots + \delta_{x,p_x}$, $\gamma = \sum_{x\in[p_0]} p_x\gamma_x$, for all $x \in [p_0]$. Let $GF(q)$ be a finite field such that*

$q \geq \max_{x\in[p_0],i\in[p_x]} \{n_{x,i} + \delta_x - (p_x - 2)\gamma_x + \gamma\}$.

Let $u_{x,i} = k_{x,i} + \delta_{x,i} + 2\gamma_x$, $v_{x,i} = r_{x,i} - \delta_{x,i} + \delta_x - p_x\gamma_x + \gamma$, *for* $x \in [p_0]$, $i \in [p_x]$. *For each* $x \in [p_0]$, $i \in [p_x]$, *let* $a_{x,i,s}, b_{x,i,t}$, $s \in [u_{x,i}]$, $t \in [v_{x,i}]$, *be distinct elements of* $\mathrm{GF}(q)$.

Consider the Cauchy matrix $\mathbf{T}_{x,i}$ *on* $\mathrm{GF}(q)^{u_{x,i}\times v_{x,i}}$ *such that* $\mathbf{T}_{x,i} = \mathbf{Y}(a_{x,i,1}, \ldots, a_{x,i,u_{x,i}}; b_{x,i,1}, \ldots, b_{x,i,v_{x,i}})$, *for* $x \in [p_0]$, $i \in [p_x]$. *Then, we obtain* $\mathbf{A}_{x,x;i,i}$, $\mathbf{B}_{x,x;i,i'}$, $\mathbf{E}_{x,y;i;j}$, $\mathbf{U}_{x,i}$, $\mathbf{V}_{x,i}$, $x \in [p_0]$, $i' \in [p_x] \setminus \{i\}$, $y \in [p_0] \setminus \{x\}$, $j \in [p_y]$, *according to the following partition of* $\mathbf{T}_{x,i}$,

$$\mathbf{T}_{x,i} = \left[\begin{array}{c|ccc} \mathbf{A}_{x,x;i,i} & \mathbf{B}_{x,x;i} & \mathbf{E}_{x,1;i} & \cdots & \mathbf{E}_{x,p_0;i} \\ \hline \begin{array}{c}\mathbf{U}_{x,i} \\ \hline \mathbf{V}_{x,i}\end{array} & \multicolumn{4}{c}{\mathbf{Z}_{x,i}} \end{array}\right], \tag{2.7}$$

$$\textit{where}\ \ \mathbf{B}_{x,x;i} = \left[\begin{array}{c|c|c} \mathbf{B}_{x,x;i,1} & \cdots & \mathbf{B}_{x,x;i,p_x} \end{array}\right] \tag{2.8}$$

$$\textit{and}\ \ \mathbf{E}_{x,y;i} = \left[\begin{array}{c|c|c} \mathbf{E}_{x,y;i;1} & \cdots & \mathbf{E}_{x,y;i;p_y} \end{array}\right], \tag{2.9}$$

such that $\mathbf{A}_{x,x;i,i} \in \mathrm{GF}(q)^{k_{x,i}\times r_{x,i}}$, $\mathbf{B}_{x,x;i,i'} \in \mathrm{GF}(q)^{k_{x,i}\times\delta_{x,i'}}$, $\mathbf{E}_{x,y;i;j} \in \mathrm{GF}(q)^{k_{x,i}\times\gamma_y}$, $\mathbf{U}_{x,i} \in \mathrm{GF}(q)^{\delta_{x,i}\times r_{x,i}}$, $\mathbf{V}_{x,i} \in \mathrm{GF}(q)^{2\gamma_x\times r_{x,i}}$. *Moreover,* $\mathbf{A}_{x,x;i,i'} = \mathbf{B}_{x,x;i,i'}\mathbf{U}_{x,i'}$. *Suppose* $\mathbf{E}_{x,y;i;p_y+1} = \mathbf{E}_{x,y;i;1}$; *let* $\mathbf{A}_{x,y;i,j} = [\mathbf{E}_{x,y;i;j}, \mathbf{E}_{x,y;i;j+1}]\,\mathbf{V}_{y,j}$.

Matrices $\mathbf{A}_{x,x;i,i}$ *and* $\mathbf{A}_{x,y;i,j}$ *are substituted in* $\mathbf{F}_{x,x}$ *and* $\mathbf{F}_{x,y}$ *to construct* $\mathbf{G}$ *as specified in (2.4), (2.5), and (2.6). Let* $\mathcal{C}_2$ *represent the code with generator matrix* $\mathbf{G}$.

Theorem 1. *Following the notation in Definition 2, let* $d_{1,x,i} = r_{x,i} - \delta_{x,i} - 2\gamma_x + 1$, $d_{2,x,i} = r_{x,i} - \delta_{x,i} + \delta_x + 1$, $d_{3,x,i} = r_{x,i} - \delta_{x,i} + \delta_x - p_x\gamma_x + \gamma + 1$, *for* $x \in [p_0]$, $i \in [p_x]$. *Then, the code* $\mathcal{C}_2$ *defined in Construction 2 is an* $(\mathbf{n}, \mathbf{k}, \mathbf{D}, p_0, \mathbf{p})_q$*-code.*

Sketch of the proof. For each $x \in [p_0]$ and $i \in [p_x]$, define the local cross parity $\mathbf{y}_{x,i} = \sum_{i'\in[p_x]\setminus\{i\}} \mathbf{m}_{x,i'}\mathbf{B}_{x,x;i,i'}$, and the global cross parities $\mathbf{z}_{x,i} = \sum_{y\in[p_0]\setminus\{x\},j\in[p_y]} \mathbf{m}_{y,j}\mathbf{E}_{y,x;j;i}$. Let $\mathbf{z}_{x,p_x+1} = \mathbf{z}_{x,p_x}$. Then, it follows from $\mathbf{mG} = \mathbf{c}$ that $\mathbf{c}_{x,i} = [\mathbf{m}_{x,i}, \mathbf{w}_{x,i}]$ for some $\mathbf{w}_{x,i} = \mathbf{m}_{x,i}\mathbf{A}_{x,x;i,i} + \mathbf{y}_{x,i}\mathbf{U}_{x,i} + [\mathbf{z}_{x,i}, \mathbf{z}_{x,i+1}]\,\mathbf{V}_{x,i}$.

The local erasure-correction capability $d_{1,x,i} = r_{x,i} - \delta_{x,i} - 2\gamma_x + 1$ and the global erasure-correction capability $d_{3,x,i} = r_{x,i} - \delta_{x,i} + \delta_x - p_x\gamma_x + \gamma + 1$ can be easily derived by following

the same logic used in the proof of Lemma 3. Therefore, we only need to prove that $d_{2,x,i} = r_{x,i} - \delta_{x,i} + \delta_x + 1$.

To prove this statement, suppose all the local codewords in the x-th group except for $\mathbf{c}_{x,i}$ are successfully decodable locally, for some $x \in [p_0]$, $i \in [p_x]$. In other words, for all $i' \in [p_x] \setminus \{i\}$, there are at most $d_{1,x,i'} - 1$ erasures in the corrupted version $\mathbf{c}_{x,i'}$ of the local codeword. From the construction, we know that the row spaces of any two matrices from $\mathbf{A}_{x,x;i,i}$, $\mathbf{U}_{x,i}$, and $\mathbf{V}_{x,i}$ have no common elements except for the all zero vector. Therefore, for all $i' \in [p_x] \setminus \{i\}$, $\mathbf{m}_{x,i'}$, $\mathbf{y}_{x,i'}$, $[\mathbf{z}_{x,i'}, \mathbf{z}_{x,i'+1}]$, can all be derived from $\mathbf{c}_{x,i}$. This implies that $[\mathbf{z}_{x,i}, \mathbf{z}_{x,i+1}]$ is known and thus, the entire contribution of global cross parities can be removed. Namely, let $\tilde{\mathbf{c}}_{x,i'} = \mathbf{c}_{x,i'} - \left[\mathbf{0}_{k_{x,i'}}, [\mathbf{z}_{x,i'}, \mathbf{z}_{x,i'+1}]\, \mathbf{V}_{x,i'}\right]$, for all $i' \in [p_x]$, then the message $\mathbf{m}_x \mathbf{F}_{x,x} = \tilde{\mathbf{c}}_x$, where $\tilde{\mathbf{c}}_x = [\tilde{\mathbf{c}}_{x,1}, \ldots, \tilde{\mathbf{c}}_{x,p_x}]$. Thus, from Lemma 3, $(r_{x,i} - \delta_{x,i} + \delta_x)$ erasures in $\tilde{\mathbf{c}}_{x,i}$ are correctable. Therefore, $d_{2,x,i} = r_{x,i} - \delta_{x,i} + \delta_x + 1$. □

Remark 1. *Note that the constraint of $\gamma_y \in \mathbb{N}$ in Construction 3 can be relaxed to $2\gamma_y \in \mathbb{N}$ if p_y is even. In this case, we have $\mathbf{E}_{x,y;i;j} \in \mathrm{GF}(q)^{k_{x,i} \times 2\gamma_y}$. Moreover, we need to modify the equation of $\mathbf{E}_{x,y;i}$ to be $\mathbf{E}_{x,y;i} = \left[\mathbf{E}_{x,y;i;1}, \ldots, \mathbf{E}_{x,y;i;p_y/2}\right]$, and $\mathbf{A}_{x,y;i,j} = \mathbf{E}_{x,y;i;\lceil j/2 \rceil} \mathbf{V}_{y,j}$.*

The following is a working example of Construction 2. For simplicity, we let the middle code be the code presented in Example 1. However, the construction itself doesn't impose any constraints on $r_{x,i}$, $\delta_{x,i}$, and γ_x, except for $2\gamma_x < \min_{y \in [p_x]} \{r_{x,y} - \delta_{x,y}\}$.

Example 2. *Here, we build on Example 1 using the same $GF(q)$. Let $p_0 = 2$, $\mathbf{p} = (p_1, p_2) = (2, 2)$, $\gamma' = \gamma_1 = \gamma_2 = 1/2$, $\gamma = p_1\gamma_1 + p_2\gamma_2 = 2$. Let $\mathbf{F}_{1,1} = \mathbf{F}_{2,2} = \mathbf{G}$ of Example 1. Then, $n = 6$, $r = 3$, $\delta' = 1$, $\delta = 2$ as in Example 1. Therefore, $d_1 = r - \delta' - 2\gamma' + 1 = 3 - 1 - 2 \cdot (1/2) + 1 = 2$, $d_2 = r - \delta' + \delta + 1 = 5$, $d_3 = r - \delta' + \delta - 2\gamma' + \gamma + 1 = 6$. We assume $\mathbf{T}_{x,i}$, $x, i \in [2]$, are all identical, then so are $\mathbf{V}_{x,i}$ and $\mathbf{E}_{x,y;i;1}$, $x \neq y$, $i \in [2]$. Let these matrices be defined as follows:*

$$\mathbf{V}_{x,i} = \begin{bmatrix} \frac{1}{\beta^6 - \beta^8} & \frac{1}{\beta^6 - \beta^9} & \frac{1}{\beta^6 - \beta^{10}} \end{bmatrix} = \begin{bmatrix} \beta & \beta^{10} & \beta^8 \end{bmatrix}$$

$$and \quad \mathbf{E}_{x,y;i;1} = \begin{bmatrix} \frac{1}{\beta-\beta^{12}} \\ \frac{1}{\beta^2-\beta^{12}} \\ \frac{1}{\beta^3-\beta^{12}} \end{bmatrix} = \begin{bmatrix} \beta^2 \\ \beta^8 \\ \beta^5 \end{bmatrix}.$$

For simplicity, we abbreviate $\mathbf{E}_{x,y;i;1}$ *as* $\mathbf{E}$*. Note that here* p_1*,* p_2 *are even; thus, the construction follows the modification described in Remark 1. The components* $\mathbf{A}_{x,y;i,j}$ *are therefore all identical for* $x, y, i, j \in [2]$*,* $x \neq y$*, and are described as follows:*

$$\mathbf{A}_{x,y;i,j} = \mathbf{E}\mathbf{V}_{y,j} = \begin{bmatrix} \beta^3 & \beta^{12} & \beta^{10} \\ \beta^9 & \beta^3 & \beta \\ \beta^6 & 1 & \beta^{13} \end{bmatrix}.$$

Then, the generator matrix is given in (2.10).

$$\left[\begin{array}{ccc|ccc|ccc|ccc|ccc|ccc|ccc} 1 & 0 & 0 & \beta^5 & \beta^{12} & \beta^7 & 0 & 0 & 0 & \beta^{13} & \beta^9 & \beta^3 & 0 & 0 & 0 & \beta^3 & \beta^{12} & \beta^{10} & 0 & 0 & 0 & \beta^3 & \beta^{12} & \beta^{10} \\ 0 & 1 & 0 & 1 & \beta^4 & \beta^{11} & 0 & 0 & 0 & \beta^{10} & \beta^6 & 1 & 0 & 0 & 0 & \beta^9 & \beta^3 & \beta & 0 & 0 & 0 & \beta^9 & \beta^3 & \beta \\ 0 & 0 & 1 & \beta^2 & \beta^{14} & \beta^3 & 0 & 0 & 0 & \beta^{14} & \beta^{10} & \beta^4 & 0 & 0 & 0 & \beta^6 & 1 & \beta^{13} & 0 & 0 & 0 & \beta^6 & 1 & \beta^{13} \\ \hline 0 & 0 & 0 & \beta^{13} & \beta^9 & \beta^3 & 1 & 0 & 0 & \beta^5 & \beta^{12} & \beta^7 & 0 & 0 & 0 & \beta^3 & \beta^{12} & \beta^{10} & 0 & 0 & 0 & \beta^3 & \beta^{12} & \beta^{10} \\ 0 & 0 & 0 & \beta^{10} & \beta^6 & 1 & 0 & 1 & 0 & 1 & \beta^4 & \beta^{11} & 0 & 0 & 0 & \beta^9 & \beta^3 & \beta & 0 & 0 & 0 & \beta^9 & \beta^3 & \beta \\ 0 & 0 & 0 & \beta^{14} & \beta^{10} & \beta^4 & 0 & 0 & 1 & \beta^2 & \beta^{14} & \beta^3 & 0 & 0 & 0 & \beta^6 & 1 & \beta^{13} & 0 & 0 & 0 & \beta^6 & 1 & \beta^{13} \\ \hline 0 & 0 & 0 & \beta^3 & \beta^{12} & \beta^{10} & 0 & 0 & 0 & \beta^3 & \beta^{12} & \beta^{10} & 1 & 0 & 0 & \beta^5 & \beta^{12} & \beta^7 & 0 & 0 & 0 & \beta^{13} & \beta^9 & \beta^3 \\ 0 & 0 & 0 & \beta^9 & \beta^3 & \beta & 0 & 0 & 0 & \beta^9 & \beta^3 & \beta & 0 & 1 & 0 & 1 & \beta^4 & \beta^{11} & 0 & 0 & 0 & \beta^{10} & \beta^6 & 1 \\ 0 & 0 & 0 & \beta^6 & 1 & \beta^{13} & 0 & 0 & 0 & \beta^6 & 1 & \beta^{13} & 0 & 0 & 1 & \beta^2 & \beta^{14} & \beta^3 & 0 & 0 & 0 & \beta^{14} & \beta^{10} & \beta^4 \\ \hline 0 & 0 & 0 & \beta^3 & \beta^{12} & \beta^{10} & 0 & 0 & 0 & \beta^3 & \beta^{12} & \beta^{10} & 0 & 0 & 0 & \beta^{13} & \beta^9 & \beta^3 & 1 & 0 & 0 & \beta^5 & \beta^{12} & \beta^7 \\ 0 & 0 & 0 & \beta^9 & \beta^3 & \beta & 0 & 0 & 0 & \beta^9 & \beta^3 & \beta & 0 & 0 & 0 & \beta^{10} & \beta^6 & 1 & 0 & 1 & 0 & 1 & \beta^4 & \beta^{11} \\ 0 & 0 & 0 & \beta^6 & 1 & \beta^{13} & 0 & 0 & 0 & \beta^6 & 1 & \beta^{13} & 0 & 0 & 0 & \beta^{14} & \beta^{10} & \beta^4 & 0 & 0 & 1 & \beta^2 & \beta^{14} & \beta^3 \end{array}\right] \tag{2.10}$$

Note that the decoding process based on local access and global access have already been introduced in Example 1. Thus, we only focus on decoding based on the middle-level access in this example. Suppose $\mathbf{m}_{1,1} = (1, \beta, \beta^2)$*,* $\mathbf{m}_{1,2} = (\beta, 1, 0)$*,* $\mathbf{m}_{2,1} = (\beta^2, 0, \beta)$*,* $\mathbf{m}_{2,2} = (0, \beta, 1)$*. Then,* $\mathbf{c}_{1,1} = (1, \beta, \beta^2, \beta^{12}, \beta^{14}, \beta^{12})$*,* $\mathbf{c}_{1,2} = (\beta, 1, 0, \beta^9, \beta^{14}, \beta)$*.*

Suppose there are 3 *erasures in* $\mathbf{c}_{1,1}$ *so that* $\mathbf{c}'_{1,1} = (e_1, \beta, \beta^2, e_2, e_3, \beta^{12})$*, where* e_1, e_2, e_3 *represent the three erased symbols. Suppose* $\mathbf{c}_{1,2}$ *is successfully corrected by local access. Then, codeword* $\mathbf{c}_{1,1}$ *is correctable through middle-level access, i.e., by operating on* $\mathbf{c}'_{1,1}$ *and* $\mathbf{c}_{1,2}$*.*

First, from $\mathbf{c}_{1,2} = (\beta, 1, 0, \beta^9, \beta^{14}, \beta)$*, we know that* $\mathbf{m}_{1,2} = (\beta, 1, 0)$*. Following the proof of Theorem 1, we know that* $(\beta^9, \beta^{14}, \beta) = \mathbf{m}_{1,2}\mathbf{A}_{1,1;1,2} + \mathbf{y}_{1,2}\mathbf{U}_{1,2} + \mathbf{z}_{1,2}\mathbf{V}_{1,2}$*. Here,* $\mathbf{y}_{1,1} = \mathbf{m}_{1,1}\mathbf{B}_{1,1;1,2}$*,* $\mathbf{z}_{1,2} = (\mathbf{m}_{2,1}+\mathbf{m}_{2,2})\mathbf{E} = \mathbf{z}_{1,1}$*. Then,* $\mathbf{y}_{1,2}$ *and* $\mathbf{z}_{1,2}$ *can be computed as* $\mathbf{y}_{1,2} = (\beta^{11})$*,* $\mathbf{z}_{1,2} = (\beta^4)$*. Therefore,* $\mathbf{z}_{1,1}\mathbf{V}_{1,1} + \mathbf{m}_{1,2}\mathbf{A}_{1,1;2,1} = \mathbf{z}_{1,2}\mathbf{V}_{1,1} + \mathbf{m}_{1,2}\mathbf{A}_{1,1;2,1} = (\beta^5, \beta^{14}, \beta^{12}) + (\beta^{11}, \beta^7, \beta) = (\beta^3, \beta, \beta^{13})$*.*

Let $\tilde{\mathbf{c}}_{1,1} = \mathbf{c}'_{1,1} - (0, 0, 0, \beta^3, \beta, \beta^{13}) = (e'_1, \beta, \beta^2, e'_2, e'_3, \beta)$*. We obtain* $(e'_1, e'_2, e'_3) = (1, \beta^{10}, \beta^7)$ *by solving* $\mathbf{H}_1^{\mathrm{G}}\tilde{\mathbf{c}}_{1,1}^{\mathrm{T}} = (0, 0, 0, e^{11})^{\mathrm{T}}$*, where* $\mathbf{H}_1^{\mathrm{G}}$ *is specified in Example 1. Therefore,* $e_1 = e'_1 = 1$*,* $e_2 = e'_2 + \beta^3 = \beta^{12}$*,* $e_3 = e'_3 + \beta = \beta^{14}$*. We have successfully decoded* $\mathbf{c}_{1,1}$*.*

2.4 Scalability, Heterogeneity, and Flexibility

In Section 2.3, we have presented a construction of codes with hierarchical locality for cloud storage, which enables the system to offer multi-level access. However, multi-level accessibility is not the only property that is considered in practical cloud storage applications. In this section, we therefore discuss scalability, heterogeneity, and flexibility of our construction, which are pivotal particularly in dynamic cloud storage. Although our discussion is restricted to cloud storage, the properties of heterogeneity and flexibility are also of practical importance in non-volatile memories.

2.4.1 Scalability

As discussed in Section 2.1, scalability refers to the capability of expanding the backbone network to accommodate additional workload without rebuilding the entire infrastructure. More specifically, when a new local cloud is added to the existing configuration, computing a completely different generator matrix resulting in changing all the encoding-decoding

components in the system is very costly. The ideal scenario is that adding a new local cloud does not change the encoding-decoding components of the already-existing, local clouds.

We show that our construction naturally achieves this goal. Observe that in Construction 3, the components $\mathbf{A}_{x,x}$, $\mathbf{U}_x$, $\mathbf{B}_{x,i}$, $i \in [p] \setminus \{x\}$ are built locally. Suppose cloud $p+1$ is added into a double-level configuration adopting Construction 3. The following steps will only result in adding some columns and rows to the original $\mathbf{G}$ without changing the existing ones:

1. *Parameter Selection:* Local cloud $p+1$ chooses its local parameters $\mathbf{A}_{p+1,p+1}$, $\mathbf{U}_{p+1}$, $\mathbf{B}_{p+1,i}$, $i \in [p]$, and local cloud i chooses the additional local parameters $\mathbf{B}_{i,p+1}$;

2. *Information Exchange:* Local cloud $p+1$ sends $\mathbf{m}_{p+1}\mathbf{B}_{p+1,i}$ to the central cloud, and local cloud i sends $\mathbf{m}_i\mathbf{B}_{i,p+1}$ to the central cloud;

3. *Information Exchange:* The central cloud forwards $\mathbf{m}_{p+1}\mathbf{B}_{p+1,i}$ to local cloud i, and sends $\mathbf{y}_{p+1} = \sum_{i\in[p]} \mathbf{m}_i\mathbf{B}_{i,p+1}$ to local cloud $p+1$;

4. *Update:* Local cloud $p+1$ computes its finalized parity-check symbols $\mathbf{m}_{p+1}\mathbf{A}_{p+1,p+1} + \mathbf{y}_{p+1}\mathbf{U}_{p+1}$, and local cloud i adds $\mathbf{m}_{p+1}\mathbf{B}_{p+1,i}$ to its current parity symbols.

Note that although the local erasure-correction capability of a local cloud does not change, the global erasure-correction capability of each local cloud increases by δ_{p+1} after adding the new local cloud $p+1$ into the system.

2.4.2 Heterogeneity

While codes with identical data length and locality have been intensively studied, heterogeneity has become increasingly important in real world applications, especially in cloud storage. There are typically two forms of heterogeneity: the heterogeneity of the network structure, and unequal usage rates (according to how hot the data stored are) of local components. It is reasonable to assume a heterogeneous structure since components connected

to a larger network are typically geographically separated and they often store data from unrelated sources. Heterogeneous networks naturally require codes with different local code lengths and nonidentical data lengths, corresponding to flexible n_x and k_x in our construction, respectively. Unequal protection of data, corresponding to flexible r_x and δ_x, also has received increasing attention in recent years. This observation is reasonable since the usage rate of the data is not necessarily identical. Clouds storing hot data (data with higher usage rate and more time urgency) should receive more local protection than those store cold data.

Although the examples we presented in Section 2.3 have identical local parameters among all the clouds for simplicity, Construction 3 and Construction 2 do not impose such restrictions, and they are actually suitable for heterogeneous configuration.

2.4.3 Flexibility

The concept of flexibility has been originally proposed and investigated for dynamic cloud storage in [22]. In a dynamic cloud storage system, the usage rate of a piece of data is not likely to remain unchanged. When the data stored in a local cloud become hot, splitting the local cloud into two smaller clouds effectively reduces the latency. However, this action should be done without reducing the erasure-correction capability of the rest of the system or changing the remaining components.

Take Construction 3 as an example, if the data stored in local cloud 1 becomes unexpectedly hot, then the following procedure splits it into two separate clouds 1^{a} and 1^{b}:

1. Select the desired local parameters $(k_1^{\mathrm{a}}, r_1^{\mathrm{a}}, \delta_1^{\mathrm{a}})$ and $(k_1^{\mathrm{b}}, r_1^{\mathrm{b}}, \delta_1^{\mathrm{b}})$ for clouds 1^{a} and 1^{b}, respectively, such that $k_1^{\mathrm{a}} + k_1^{\mathrm{b}} = k_1$, $r_1^{\mathrm{a}} + r_1^{\mathrm{b}} = r_1$, $\delta_1^{\mathrm{a}} + \delta_1^{\mathrm{b}} = \delta_1$, and

$$\begin{aligned}
\mathbf{A}_{1^{\mathrm{a}},1^{\mathrm{a}}} &= \mathbf{A}_{1,1}\left[1 : k_1^{\mathrm{a}}, 1 : r_1^{\mathrm{a}}\right], \\
\mathbf{B}_{1^{\mathrm{b}},1^{\mathrm{a}}} &= \mathbf{A}_{1,1}\left[k_1^{\mathrm{a}} + 1 : k_1, 1 : \delta_1^{\mathrm{a}}\right], \\
\mathbf{A}_{1^{\mathrm{b}},1^{\mathrm{b}}} &= \mathbf{A}_{1,1}\left[k_1^{\mathrm{a}} + 1 : k_1, , r_1^{\mathrm{a}} + 1 : r_1\right], \\
\mathbf{B}_{1^{\mathrm{a}},1^{\mathrm{b}}} &= \mathbf{A}_{1,1}\left[1 : k_1^{\mathrm{a}}, r_1^{\mathrm{a}} + 1 : r_1^{\mathrm{a}} + \delta_1^{\mathrm{b}}\right],
\end{aligned}$$

$$\mathbf{U}_1^{\mathrm{a}} = \mathbf{U}_1 \left[1 : \delta_1^{\mathrm{a}}, 1 : r_1^{\mathrm{a}}\right],$$
$$\mathbf{U}_1^{\mathrm{b}} = \mathbf{U}_1 \left[\delta_1^{\mathrm{a}} + 1 : \delta_1, r_1^{\mathrm{a}} + 1 : r_1^{\mathrm{b}}\right];$$

2. Compute $\mathbf{y}_1$ by solving the equation $\mathbf{y}_1\mathbf{U}_1 = \mathbf{c}_1 - \mathbf{m}_1\mathbf{A}_{1,1}$, where $\mathbf{y}_i$, $i \in [p]$, are described in the proof of Lemma 3. Find $\mathbf{y}_1^{\mathrm{a}} \in \mathrm{GF}(q)^{\delta_1^{\mathrm{a}}}$, $\mathbf{y}_1^{\mathrm{b}} \in \mathrm{GF}(q)^{\delta_1^{\mathrm{b}}}$ such that $\mathbf{y}_1 = \left[\mathbf{y}_1^{\mathrm{a}}, \mathbf{y}_1^{\mathrm{b}}\right]$;

3. Compute $\mathbf{c}_1^{\mathrm{a}} = \left[\mathbf{m}_1^{\mathrm{a}}, \mathbf{m}_1^{\mathrm{a}}\mathbf{A}_{1^{\mathrm{a}},1^{\mathrm{a}}} + \left(\mathbf{m}_1^{\mathrm{b}}\mathbf{B}_{1^{\mathrm{b}},1^{\mathrm{a}}} + \mathbf{y}_1^{\mathrm{a}}\right)\mathbf{U}_1^{\mathrm{a}}\right]$, and
$\mathbf{c}_1^{\mathrm{b}} = \left[\mathbf{m}_1^{\mathrm{b}}, \mathbf{m}_1^{\mathrm{b}}\mathbf{A}_{1^{\mathrm{b}},1^{\mathrm{b}}} + \left(\mathbf{m}_1^{\mathrm{a}}\mathbf{B}_{1^{\mathrm{a}},1^{\mathrm{b}}} + \mathbf{y}_1^{\mathrm{b}}\right)\mathbf{U}_1^{\mathrm{b}}\right]$.

One can prove that the local codewords stored in the new clouds 1^{a} and 1^{b} such that they are capable of correcting $(r_1^{\mathrm{a}} - \delta_1^{\mathrm{a}})$ and $(r_1^{\mathrm{b}} - \delta_1^{\mathrm{b}})$ local erasures, respectively. Other local clouds are not affected.

2.5 Conclusion

Multi-level accessible codes have been shown to be beneficial for cloud storage. While the previous literature works was typically focused on double-level accessible codes and their erasure-correction capabilities, in this chapter, we focus on codes with hierarchical locality and additional properties motivated by their practical importance. We proposed a CRS-based code on a finite field with size that grows linearly with the maximum local codelength. We showed that our construction achieves scalability, heterogeneity and flexibility, which are important in dynamic cloud storage.

CHAPTER 3

Hierarchical Coding for Decentralized Cloud Storage

3.1 Introduction

In response to the rapidly growing demand of data management, cloud storage such as Microsoft Azure and Amazon Web Services have become among the most widely deployed public cloud services. In these centralized cloud services, a tech giant takes full custodianship over data of all its customers; this situation can result in expensive infrastructure maintenance and may lead to privacy violations. Decentralized storage networks (DSNs) such as Storj [18], in which no entity is solely responsible for all data, have emerged as a secure and economic alternative to centralized cloud services. DSNs are believed to be economically attractive since extra capacity can be afforded by utilizing idle storage space on devices at the edge of the network. Despite all advantages of decentralization, practical management of personal devices also faces challenges from component failures, high churn rates, heterogeneous bandwidths and link speeds, in addition to dynamic node balancing for content delivery of hot files [18]. While erasure correction (EC) codes are widely used to combat component failures, EC schemes that address the aforementioned issues are relatively overlooked. In this chapter, we propose EC solutions that are tailored to tackle those challenges pertaining to DSNs.

Latency and reliability are among the most critical factors that customers care about in cloud storage. However, DSNs naturally impose numerous challenges on simultaneously

maintaining low latency and high reliability. EC solutions with large block lengths are more resilient to large weight errors, but they simultaneously slow down the recovery for the more frequent cases where only few erasures occur. To reach a better trade-off between data reliability and latency, codes enabling multi-level access are desired. In these codes, any node is allowed to access different sets of helper nodes to retrieve the data, where the sizes of the sets get reduced if the number of erasures to be recovered is small enough. This architecture is referred to as codes with hierarchical localities. While hierarchical coding in the context of centralized storage [9, 10, 11, 12, 13, 14, 15, 16, 33] has been intensively studied, codes for DSNs have been mostly discussed without considering localities [1, 65, 66].

More recently, codes with localities in multi-rack storage, a special case of DSNs, have also been investigated, where either the system is considered to be homogeneous [29, 30, 31, 32], or the network topology has a simple structure [28, 23]. However, DSNs typically have more sophisticated topologies characterized by heterogeneity among bandwidths of communication links and erasure statistics of nodes due to the arbitrary and dynamic nature of practical networks [23, 24, 25, 26, 27]. Instead of solutions for simplified models, schemes that fit into any topology (a property referred to as **topology-adaptivity** later on) with customizable data lengths and redundancies are desired to exploit the existing resources.

Another major challenge for DSNs comes from the high churn rate, namely, participants join the network and leave without a predictable pattern. Therefore, it is essential for a DSN to enable its organic growth, i.e., enable expanding the backbone network to accommodate additional node operators, without rebuilding the entire infrastructure [17]; this property is referred to as **scalability**. It has been reported that Storj has around 13500 storage nodes across its global network, where each file is split into 80 pieces and any 29 of them is needed to recover a file [67]. This indicates that the network is of massive scale and the erasure correction is indispensable. Moreover, data stored at certain nodes occasionally become hotter than anticipated, and the download rate can thus exceed the bandwidth limit. In such a scenario, dynamic node balancing is required for content delivery to reach

a lower latency. In particular, the cloud (node) should be split into smaller clouds without worsening the global erasure correction capability or changing the remaining components. This property is referred to as **flexibility** and has been firstly investigated for dynamic data storage systems under the discussion of the so-called sum-rank codes [22]. However, sum-rank codes require a Galois field size that grows *exponentially* with the maximum local block length, which is a major obstacle to being implemented in real world applications [22].

In this chapter, we strategically combine hierarchical locality and topological properties of a DSN. We develop a hierarchical coding scheme that is topologically-adaptive. The scheme is built upon our prior work on centralized cloud storage [33] and preserves desirable properties including scalability and flexibility. The Galois field size of this scheme grows *linearly* with the local block length. Our proposed coding scheme enables joint encoding and decoding of the data stored at all nodes such that nodes in a neighborhood cooperatively protect and validate their stored data collectively in the DSN. Cooperation in DSNs further improves the reliability since information propagates from more reliable nodes to less reliable nodes through paths connecting them.

The rest of the chapter is organized as follows. In Section 3.2, we introduce the DSN model and necessary preliminaries. In Section 3.3, we define erasure correction (EC) hierarchies as well as their depth to systematically describe the maximal number of recoverable erasures corresponding to different access levels. We present a coding scheme with depth 1 that results in a better recovery speed compared with existing schemes that are not topologically-adaptive [28, 23]. We also discuss the recoverable erasure patterns of the proposed construction and show that our scheme enables correction of erasure patterns relevant to DSNs. In Section 3.4, we extend the single-level construction (depth 1) to have higher-level cooperation. In the hierarchical scheme, the cooperation between nodes in the DSN is described by the so-called **cooperation graphs**. We also present sufficient conditions on any graph to be a cooperation graph, and refer to graphs satisfying these conditions as **compatible graphs**. In Section 3.5, we first present an algorithm that searches for a

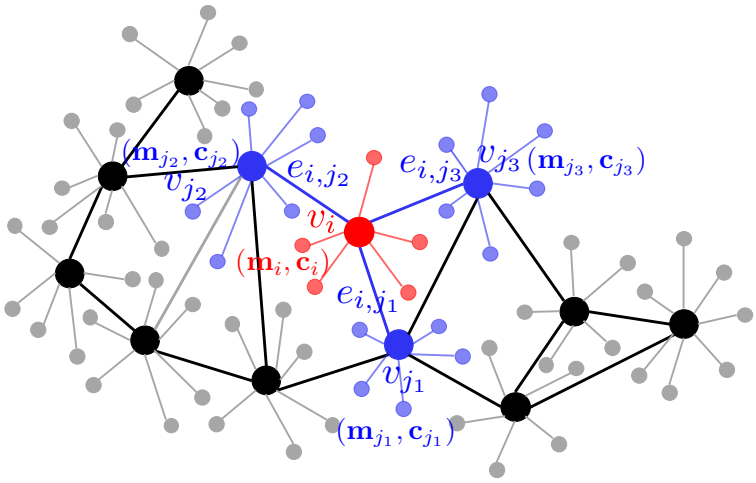


Figure 3.1: Decentralized storage network (DSN). For the cluster with the master node v_i, message $\mathbf{m}_i$ is encoded to $\mathbf{c}_i$, and symbols of $\mathbf{c}_i$ are stored distributively among non-master nodes that are locally connected to v_i. In the figures after Fig. 3.1, we omit the local non-master nodes for clarity of figures.

cooperation graph on any DSN with a given topology. Next, we show that our coding scheme supports scalability and flexibility. Finally, we summarize our results in Section 3.6.

3.2 Notation and Model

In this section, we discuss the model and mathematical representation of a DSN, as well as necessary preliminaries. Throughout the remainder of this chapter, $[N]$ refers to $\{1, 2, \ldots, N\}$. For a vector $\mathbf{v}$ of length n, v_i, $1 \leq i \leq n$, represents the i-th component of $\mathbf{v}$, and $\mathbf{v}[a:b] = (v_a, \ldots, v_b)$. For a matrix $\mathbf{M}$ of size $a \times b$, $\mathbf{M}[i_1 : i_2, j_1 : j_2]$ represents the sub-matrix $\mathbf{M}'$ of $\mathbf{M}$ such that $(\mathbf{M}')_{i-i_1+1,j-j_1+1} = (\mathbf{M})_{i,j}$, $i \in [i_1 : i_2]$, $j \in [j_1 : j_2]$. For vectors $\mathbf{u}$ and $\mathbf{v}$ of the same length p, $\mathbf{u} \succ \mathbf{v}$ and $\mathbf{u} \prec \mathbf{v}$ means $u_i > v_i$ and $u_i < v_i$, for all $i \in [p]$, respectively; $\mathbf{u} \succeq \mathbf{v}$ and $\mathbf{u} \preceq \mathbf{v}$ means $u_i \geq v_i$ and $u_i \leq v_i$, for all $i \in [p]$, respectively. For any $m, n \in \mathbb{N}$, an identity matrix of size $n \times n$ is denoted by $\mathbf{I}_n$, and a zero matrix of size $m \times n$ is denoted by $\mathbf{0}_{m \times n}$. For any $q \in \mathbb{N}$, GF(q) refers to a Galois field with size q. In this chapter, we constrain q to be a power of 2.

3.2.1 Decentralized Storage Network

In a DSN, nodes are typically partitioned into distributed clusters of nodes, where each cluster has a "master node" that functions in this cluster similar to that of a central node in a centralized network, which is what the "decentralization" refers to. As shown in Fig. 3.1, each master node, represented by big bold-colored nodes, communicates with both its neighboring master nodes and other nodes in the cluster it belongs to, whereas each non-master node, represented by small light-colored nodes, only communicates with the master node of the cluster it belongs to. The cost of intra-cluster communications is negligible compared with the cost of inter-cluster communications. Therefore, for clarity and simplicity of figures and notation, it is sufficient to use a single compound node to represent each cluster in the remainder of the paper, where each compound node accesses the storage nodes in the represented cluster and performs calculations over the data. Moreover, the communications between master nodes in clusters can be simply referred to as the communications between the compound nodes representing them. We then refer to the compound nodes and the communication links among them as "nodes" and "edges", respectively.

As shown in Fig. 3.1, a DSN is modeled as a graph $G(V, E)$, where V and E denote the set of nodes and edges, respectively. Let $p = |V|$, G is associated with a tuple $(\mathbf{n}, \mathbf{k}, \mathbf{r}) \in (\mathbb{N}^p)^3$, where $\mathbf{k}, \mathbf{r} \succ \mathbf{0}$ and $\mathbf{n} = \mathbf{k}+\mathbf{r}$. For the cluster represented by node $v_i \in V$, $1 \leq i \leq p$, message $\mathbf{m}_i \in \mathrm{GF}(q)^{k_i}$ is encoded to $\mathbf{c}_i \in \mathrm{GF}(q)^{n_i}$, and symbols of $\mathbf{c}_i$ are stored distributively among storage nodes in the cluster represented by v_i. Failed components in a cluster are regarded as erased symbols in the codeword stored at this cluster. For simplicity, we instead say "$\mathbf{c}_i$ is stored at node v_i" in the rest of the paper. Each edge $e_{i,j} \in E$, $1 \leq i, j \leq p$ and $i \neq j$, represents a communication link connecting node v_i and node v_j, through which v_i and v_j are allowed to exchange information. Messages $\{\mathbf{m}_i\}_{v_i \in V}$ are jointly encoded as $\{\mathbf{c}_i\}_{v_i \in V}$, and $\mathbf{c}_i$ is stored at the cluster of nodes containing v_i.

Each node v_i, $1 \leq i \leq p$, is associated with $L_i \in \mathbb{N}$, vector $\mathbf{d}_i = (d_{i,0}, d_{i,1}, \ldots, d_{i,L_i}) \in$

$\mathbb{N}^{L_i+1}$, and sets $\{\mathcal{H}_i^\ell\}_{\ell=1}^{L_i}$, where $d_{i,0} < r_i < d_{i,1} < \cdots < d_{i,L_i}$ and $\varnothing \subset \mathcal{H}_i^1 \subset \mathcal{H}_i^2 \subset \cdots \subset \mathcal{H}_i^{L_i} \subseteq V$, for all $\ell \in [L_i]$. The maximum number of erased symbols v_i can recover in $\mathbf{c}_i$ locally, i.e., without communicating with neighboring nodes, is $d_{i,0}$. Moreover, $d_{i,\ell}$ represents the maximum number of erased symbols v_i can recover in $\mathbf{c}_i$ if codewords stored on all nodes from $\mathcal{H}_i^\ell$ are recoverable, for all $\ell \in [L_i]$. This property is referred to later as **hierarchical locality** and discussed later in detail. Let $\mathbf{m} = (\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_p)$ and $\mathbf{c} = (\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_p)$. Denote the set of all neighbors of node v_i by $\mathcal{N}_i$, e.g., $\mathcal{N}_i = \{v_{j_1}, v_{j_2}, v_{j_3}\}$ in Fig. 3.1, and refer to it as the **neighborhood** of node v_i. A coding scheme $\mathcal{C} : \otimes_{i=1}^p \mathrm{GF}(q)^{k_i} \to \otimes_{i=1}^p \mathrm{GF}(q)^{n_i}$ maps each vector $\mathbf{m}$ of messages to a vector $\mathbf{c}$ of codewords. As mentioned previously, our major goal is to design coding solutions $\mathcal{C}$ that are topology-adaptive, scalable, and flexible. The aforementioned properties are specified as follows:

1. **Topology-adaptivity**: The selection of the sets of helper nodes $\{\mathcal{H}_i^\ell\}_{i=1}^\ell$ is based on the underlying topology G of the DSN, which specifies the graph indicating the information flow, the so-called **cooperation graph** discussed in details later on. The code construction is based on the chosen cooperation graph.

   In particular, $\mathcal{H}_i^1$ only contains the neighborhood $\mathcal{N}_i$ of v_i and the union of the neighborhoods of nodes in $\mathcal{N}_i$. Node v_i is able to correct r_i erasures in $\mathbf{c}_i$ (i.e., fully utilizes its local redundancy) when codewords stored on nodes from its neighborhood are recoverable, and it can receive additional parities (up to $(d_{i,1} - r_i)$ parities) from nodes in neighborhood of $v_j \in \mathcal{N}_i$ through v_j if v_j is further recovered.

   In addition to that, while the information of helper nodes participating in higher level cooperations are encoded and stored in v_i, the construction must ensure that the information from nodes of higher level cooperations can be removed even if they are not recovered, which is required in lower level cooperations. Simultaneously, the redundancy/erasure correction capability trade-off cannot be compromised. In particular, the construction should accommodate a large set of erasure patterns where the sum of erasures in each node is equal to the number of redundant symbols in the network.

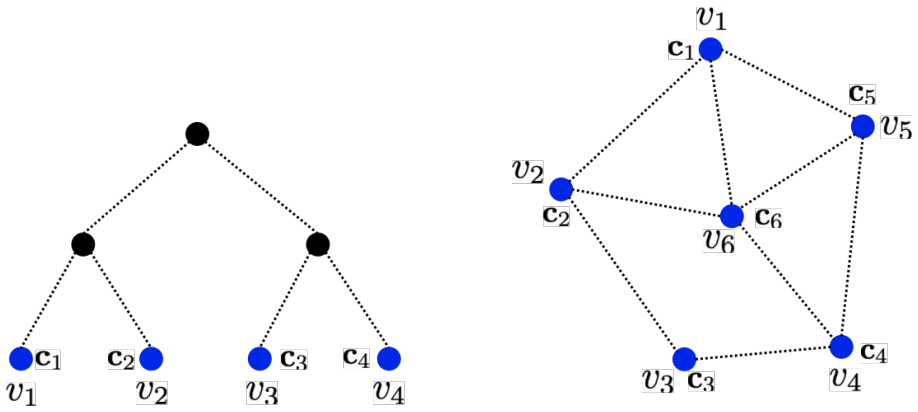


Figure 3.2: Examples of a hierarchically centralized network (a) and a DSN (b).

Note that the selection of the cooperation graph and constructions of the coding scheme over it satisfying the above properties is nontrivial, and will be discussed later in detail.

2. **Scalability**: When adding a node v_{p+1} with local parameters $(n_{p+1}, k_{p+1}, r_{p+1}) \in \mathbb{N}^3$ that encodes $\mathbf{m}_{p+1}$, v_{p+1} is able to compute the codeword $\mathbf{c}_{p+1}$ to be stored in it locally with limited communications with other nodes, and without accessing the encoding functions of other nodes nor changing codewords stored on nodes from $V \setminus \mathcal{H}_{p+1}^{L_{p+1}}$.

3. **Flexibility**: When a node v_i becomes hotter than anticipated, both the message $\mathbf{m}_i$ and the codeword $\mathbf{c}_i$ can be split into two shorter components, namely, $\mathbf{m}_i = (\mathbf{m}_{i^{\mathrm{a}}}, \mathbf{m}_{i^{\mathrm{b}}})$ and $\mathbf{c}_i = (\mathbf{c}_{i^{\mathrm{a}}}, \mathbf{c}_{i^{\mathrm{b}}})$. Then, $\mathbf{c}_{i^{\mathrm{a}}}$, $\mathbf{c}_{i^{\mathrm{b}}}$ can be stored at two smaller nodes $v_{i^{\mathrm{a}}}$ and $v_{i^{\mathrm{b}}}$ such that $\mathbf{m}_{i^{\mathrm{a}}}$, $\mathbf{m}_{i^{\mathrm{b}}}$ can be decoded from $\mathbf{c}_{i^{\mathrm{a}}}$ and $\mathbf{c}_{i^{\mathrm{b}}}$ with $d_{i,0} = d_{i^{\mathrm{a}},0} + d_{i^{\mathrm{b}},0}$. Moreover, knowledge about encoding functions of other nodes is not required; the codewords stored on nodes from $V \setminus \mathcal{H}_i^{L_i}$ are not changed.

The major difference between a centralized cloud and a DSN is that the DSN requires codes that adapt to arbitrary topologies. In particular, as shown in Fig. 3.2(a), a centralized network can be represented as a tree, where only the leaves are storage nodes and other nodes are virtual nodes indicating different decoding layers. In particular, each virtual node can access the content of all its descendants. From the bottom of the tree to its top, the number

of tolerable erasures increases. In contrast, in the DSN shown in Fig. 3.2(b), nodes only communicate with their neighbors, which requires more subtly designed codes to enable a decoding solution that strictly fits into the topology. Therefore, the existing literature [9, 10, 11, 12, 13, 14] works on locally recoverable codes can not be directly applied to DSNs.

While Cauchy matrices have been applied in our previous construction for centralized clouds [33], these constructions cannot be directly applied to DSNs. Among the three properties we discussed above, scalability and flexibility are inherited from the constructions for centralized clouds: they are achieved, in part, because of the underlying Cauchy matrices. Topology-adaptivity, on the other hand, is a property unique to constructions for DSNs, and it is discussed here under the scenario where whether two nodes can communicate depends on practical relevance characterized by metrics such as their geographical distances. One major difficulty in code constructions for DSNs is that while each node stores information about its neighboring and some non-neighboring nodes due to higher-level cooperations, the information should be strategically placed and combined such that each node is still able to remove this information at lower-level cooperations. This challenge is easier to solve in hierarchically centralized cloud networks due to their simple topologies. On the contrary, this challenge becomes more difficult in DSNs given the complicated topologies that make the nodes more intricately correlated.

3.2.2 Locality of Interleaved Cauchy Reed Solomon Codes

A code is **systematic** if the codeword contains a segment that is identical to the message being encoded. For a linear block code, systematic encoding of messages with length k is performed via a generator matrix containing a $k \times k$ submatrix being the identity matrix $\mathbf{I}_k$. Systematic codes are of interest because of their low complexity mapping from any valid codeword to the message it represents, as well as their low encoding complexity due to the fact that only parities need extra calculation steps. Based on the aforementioned notation,

a systematic generator matrix of a code on $G(V, E)$ has the following structure:

$$\mathbf{G} = \left[\begin{array}{c|c|c|c|c|c|c} \mathbf{I}_{k_1} & \mathbf{A}_{1,1} & \mathbf{0} & \mathbf{A}_{1,2} & \dots & \mathbf{0} & \mathbf{A}_{1,p} \\ \hline \mathbf{0} & \mathbf{A}_{2,1} & \mathbf{I}_{k_2} & \mathbf{A}_{2,2} & \dots & \mathbf{0} & \mathbf{A}_{2,p} \\ \hline \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \hline \mathbf{0} & \mathbf{A}_{p,1} & \mathbf{0} & \mathbf{A}_{p,2} & \dots & \mathbf{I}_{k_p} & \mathbf{A}_{p,p} \end{array}\right], \tag{3.1}$$

where all elements are from a Galois field $\mathrm{GF}(q)$, $q = 2^\theta$ and $\theta \geq 2$.

Following the notation in the previous subsections, the codeword at node v_i is $\mathbf{c}_i = (\mathbf{m}_i, \sum_{j\in[p]} \mathbf{m}_j \mathbf{A}_{j,i})$, and it has two parts. We call $\mathbf{m}_i$ the systematic part, and $\sum_{j\in[p]} \mathbf{m}_j \mathbf{A}_{j,i}$ the **local parities** of $\mathbf{c}_i$. More specifically, we refer to $\sum_{j\in[p],j\neq i} \mathbf{m}_j \mathbf{A}_{j,i}$, $\mathbf{m}_i \mathbf{A}_{i,i}$ as the **additional local parities** and the **original local parities** at v_i, respectively. For any $j \in [p]$, $j \neq i$, symbols in $\mathbf{m}_i \mathbf{A}_{i,j}$ are referred to as the **cross parities** of v_i from node v_j. Note that by saying "parities" we actually mean "parity symbols". We use these two terms interchangeably in the remaining text.

The submatrices $\{\mathbf{A}_{i,j}\}_{i,j\in[p]}$ in our construction are either zero matrices, Cauchy matrices, or products of Cauchy matrices. For this reason, we call codes represented by a generator matrix in (3.1) as **interleaved Cauchy Reed Solomon (CRS) codes**. The primary property of interleaved CRS codes is that each local message $\mathbf{m}_i$ is to be obtained locally by only accessing the codeword $\mathbf{c}_i$ stored at v_i if the number of erasures in $\mathbf{c}_i$ does not exceed an upper bound that is determined by some local parameters. We next provide an exemplary construction, Construction 3, to illustrate the locality of interleaved CRS codes.

Construction 3. *(Interleaved CRS codes) Let $p \in \mathbb{N}$, $k_1, k_2, \dots, k_p \in \mathbb{N}$, $n_1, n_2, \dots, n_p \in \mathbb{N}$, $\delta_1, \delta_2, \dots, \delta_p \in \mathbb{N}$, with $r_x = n_x - k_x > 0$ for all $x \in [p]$. Let $P = ([p] \times [p]) \setminus \{(i,i)\}_{i\in[p]}$, and $I \subseteq P$ is such that for all $(x,y) \in I$, $\mathbf{A}_{x,y}$ is non-zero. Let $I_x = \{i : (x,i) \in I\}$, for each $x \in [p]$, and suppose $I_x = \{i_1, i_2, \cdots, i_{|I_x|}\}$. Let $\delta'_x = \sum_{y\in I_x} \delta_y$, for all $x \in [p]$. Let $\mathrm{GF}(q)$ be a Galois field such that $q \geq \max_{x\in[p]}\{n_x + \delta'_x\}$.*

For each $x \in [p]$, let $a_{x,i}$, $i \in [k_x + \delta_x]$, and $b_{x,j}$, $j \in [r_x - \delta_x + \delta'_x]$, be distinct el-

ements of $\mathrm{GF}(q)$. *Consider the Cauchy matrix* $\mathbf{T}_x \in \mathrm{GF}(q)^{(k_x+\delta_x)\times(r_x-\delta_x+\delta'_x)}$ *such that* $\mathbf{T}_x = \mathbf{Y}(a_{x,1}, \ldots, a_{x,k_x+\delta_x}; b_{x,1}, \ldots, b_{x,r_x-\delta_x+\delta'_x})$. *For each* $x \in [p]$, *we obtain* $\{\mathbf{B}_{x,i}\}_{i\in I_x}$, $\mathbf{U}_x$, $\mathbf{A}_{x,x}$, *according to the following partition of* $\mathbf{T}_x$*:*

$$\mathbf{T}_x = \left[\begin{array}{c|c|c|c} \mathbf{A}_{x,x} & \mathbf{B}_{x,i_1} & \ldots & \mathbf{B}_{x,i_{|I_x|}} \\ \hline \mathbf{U}_x & \multicolumn{3}{c}{\mathbf{Z}_x} \end{array}\right], \tag{3.2}$$

where $\mathbf{A}_{x,x} \in \mathrm{GF}(q)^{k_x\times r_x}$, $\mathbf{B}_{x,i} \in \mathrm{GF}(q)^{k_x\times\delta_i}$, $\mathbf{U}_x \in \mathrm{GF}(q)^{\delta_x\times r_x}$. *Moreover, let* $\mathbf{A}_{x,y} = \mathbf{B}_{x,y}\mathbf{U}_y$, *for* $(x,y) \in I$; *let* $\mathbf{A}_{x,y} = \mathbf{0}_{k_x\times r_y}$, *for* $(x,y) \in P \setminus I$.

Matrices $\mathbf{A}_{x,x}$ *and* $\mathbf{A}_{x,y}$ *are substituted in* $\mathbf{G}$ *specified in (3.1), for all* $x, y \in [p]$. *Let* $\mathcal{C}_1$ *represent the code with the generator matrix* $\mathbf{G}$.

Following the notation in Subsection 3.2.1, suppose in a DSN that is implemented with a code specified in Construction 3, all nodes are able to communicate with each other. For all $x \in [p]$, let $d_{x,1}$, $d_{x,2}$ represent the maximum number of erasures that node v_x can tolerate with local access to the codeword $\mathbf{c}_x$, and global access to all the codewords $\{\mathbf{c}_x\}_{x\in[p]}$, respectively. Lemma 3 presents the value of the local and the global correction capabilities of codes proposed in Construction 3. Note that even though $\mathbf{m}_j\mathbf{A}_{j,i} = \mathbf{m}_j\mathbf{B}_{j,i}\mathbf{U}_i$ gives the explicit cross parities, symbols resulting from $\mathbf{m}_j\mathbf{B}_{j,i}$ can accurately be seen as the cross parities too since they constitute a set of independent linear combinations of message symbols, and they contain all the information node v_j provides to node v_i, for all $v_i, v_j \in V, i \neq j$. Therefore, in the remainder of this chapter, we also refer to $\mathbf{m}_j\mathbf{B}_{j,i}$ as the cross parities or the cross parity symbols for simplicity.

Lemma 3. *In code* $\mathcal{C}_1$ *specified in Construction 3,* $d_{x,1} = r_x - \delta_x$, $d_{x,2} = r_x + \delta'_x$, *for* $x \in [p]$.

Proof. For each $x \in [p]$, define $\mathbf{y}_x = \sum_{y\in I_x} \mathbf{m}_y\mathbf{B}_{y,x}$. It follows from $\mathbf{mG} = \mathbf{c}$ and (3.1) that for $x \in [p]$, $\mathbf{c}_x = [\mathbf{m}_x, \mathbf{m}_x\mathbf{A}_{x,x} + \mathbf{y}_x\mathbf{U}_x]$. Define the local parity-check matrix $\mathbf{H}_x^{\mathrm{L}}$ and the

Table 3.1: Polynomial and binary representation of GF(2^4)

0	0000	β^4	1100	β^8	1010	β^{12}	1111
β	0100	β^5	0110	β^9	0101	β^{13}	1011
β^2	0010	β^6	0011	β^{10}	1110	β^{14}	1001
β^3	0001	β^7	1101	β^{11}	0111	$\beta^{15} = 1$	1000

global parity-check matrix $\mathbf{H}_x^{\mathrm{G}}$, for each $x \in [p]$, as follows:

$$\mathbf{H}_x^{\mathrm{G}} = \left[\begin{array}{c|c|c|c} \mathbf{A}_{x,x} & \mathbf{B}_{x,i_1} & \cdots & \mathbf{B}_{x,i_{|I_x|}} \\ \hline -\mathbf{I}_{r_x} & \multicolumn{3}{c}{\mathbf{0}_{r_x \times \delta'_x}} \end{array}\right]^{\mathrm{T}}, \quad \mathbf{H}_x^{\mathrm{L}} = \begin{bmatrix} \mathbf{A}_{x,x} \\ -\mathbf{I}_{r_x} \\ \mathbf{U}_x \end{bmatrix}^{\mathrm{T}}.$$

We next prove the equations of the local correction capability $d_{x,1} = r_x - \delta_x$ and the global correction capability $d_{x,2} = r_x + \delta'_x$ using $\mathbf{H}_x^{\mathrm{L}}$ and $\mathbf{H}_x^{\mathrm{G}}$, $x \in [p]$.

To prove the equation of the local correction capability, let $\tilde{\mathbf{c}}_x = [\mathbf{c}_x, \mathbf{y}_x]$. Then, one can show that $\tilde{\mathbf{c}}_x$ belongs to a code $\mathcal{C}_x^{\mathrm{L}}$ with the local parity-check matrix $\mathbf{H}_x^{\mathrm{L}}$. From Lemma 1, $\mathcal{C}_x^{\mathrm{L}}$ is an $(n_x + \delta_x, k_x, r_x + 1)_q$-code. Therefore, any r_x erasures in $\tilde{\mathbf{c}}_x$ are correctable. Provided that $\mathbf{y}_x$ has length δ_x, we can consider the entries of $\mathbf{y}_x$ as erasures, and thus any $(r_x - \delta_x)$ erasures in the remaining part of $\tilde{\mathbf{c}}_x$, i.e., $\mathbf{c}_x$, can be corrected. Therefore, $d_{x,1} = r_x - \delta_x$.

To prove the equation of the global correction capability, assume all the local codewords except for $\mathbf{c}_x$ are successfully decodable locally. For each $x \in [p]$, let $\mathbf{s}_x = \left[\mathbf{m}_x \mathbf{B}_{x,i_1}, \ldots, \mathbf{m}_x \mathbf{B}_{x,i_{|I_x|}}\right]$ and $\bar{\mathbf{c}}_x = \mathbf{c}_x - [\mathbf{0}_{k_x}, \mathbf{y}_x \mathbf{U}_x]$. Then, one can show that $\mathbf{H}_x^{\mathrm{G}} \bar{\mathbf{c}}_x^{\mathrm{T}} = [\mathbf{0}_{r_x}, \mathbf{s}_x]^{\mathrm{T}}$. From Lemma 1 and from the construction of $\mathbf{H}_x^{\mathrm{G}}$, any $(r_x + \delta'_x)$ erasures in $\bar{\mathbf{c}}_x$ are correctable, and thus $(r_x + \delta'_x)$ erasures in $\mathbf{c}_x$ are also correctable. Therefore, $d_{x,2} = r_x + \delta'_x$. □

Note that Construction 3 is proposed based on the assumption that any node is able to communicate with all the nodes in the network; namely, the underlying DSN has a specific topology embodied in a complete graph. However, as discussed in Section 4.1 and shown in Fig. 3.1, practical DSNs are not necessarily constrained into any specific structures. A major reason is that nodes are typically scattered in geographically separated locations and

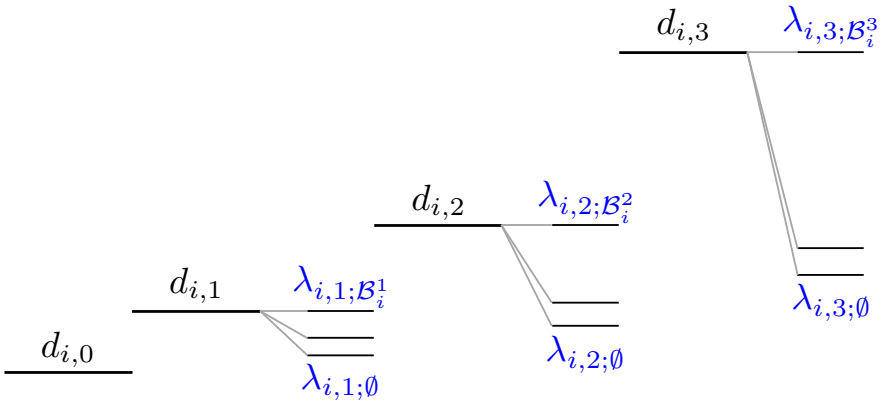


Figure 3.3: EC hierarchy of node $v_i \in V$. The values $d_{i,0}$ and $d_{i,\ell}$, $1 \leq \ell \leq 3$, represent the maximum number of erasures v_i can tolerate in the local decoding, and decoding with the assistance of the 1-st to the ℓ-th level cooperations of the codeword $\mathbf{c}_i$, i.e., cooperating with nodes in $\mathcal{A}_i^\ell$, respectively. However, even for a fixed $\mathcal{A}_i^\ell$, different sets $\mathcal{W}$ of nodes that are recovered in $\mathcal{B}_i^\ell$ may also result in different EC capabilities; we refer to them as $(\lambda_{i,\ell;\mathcal{W}})$, $\varnothing \subseteq \mathcal{W} \subseteq \mathcal{B}_i^\ell$.

communicate with only a few nodes nearby. Even if connections of nodes are not determined by physical locations, their logical connections can still be of any topology tailored for particular requirements from users of the services those nodes provide. Therefore, it is important to generalize our previous construction into one that fits into arbitrary topologies. In the next section, we take network topology into account and focus on constructions that are topology-adaptive.

3.3 Cooperative Data Protection

In this section, we first mathematically describe the EC hierarchy and its depth associated with the given DSN. EC hierarchy specifies the EC capabilities of nodes while cooperating with different sets of other nodes. We then propose a cooperation scheme where each node only cooperates with its single-hop neighbors.

3.3.1 EC Hierarchy

Denote the **EC hierarchy** of node $v_i \in V$ by a sequence $\mathbf{d}_i = (d_{i,0}, d_{i,1}, \ldots, d_{i,L_i})$, where L_i is called the **depth** of $\mathbf{d}_i$, and $d_{i,\ell}$ represents the maximum number of erased symbols v_i can recover in its codeword $\mathbf{c}_i$ from the ℓ-th level cooperation, for all $\ell \in [L_i]$. The maximum number of erased symbols v_i can recover in $\mathbf{c}_i$ locally, i.e., without communicating with neighboring nodes, is $d_{i,0}$.

For each $v_i \in V$ such that $L_i > 0$, there exist two series of sets of nodes, denoted by $\varnothing \subset \mathcal{A}_i^1 \subset \mathcal{A}_i^2 \subset \cdots \subset \mathcal{A}_i^{L_i} \subseteq V$ and $\{\mathcal{B}_i^\ell\}_{\ell=1}^{L_i}$, where $\mathcal{A}_i^\ell \cap \mathcal{B}_i^\ell = \varnothing$ for all $\ell \in [L_i]$, and a series $(\lambda_{i,\ell;\mathcal{W}})_{\varnothing \subseteq \mathcal{W} \subseteq \mathcal{B}_i^\ell}$. In the ℓ-th level cooperation, node $v_i \in V$ tolerates $\lambda_{i,\ell;\mathcal{W}}$ ($\varnothing \subseteq \mathcal{W} \subseteq \mathcal{B}_i^\ell$) erasures if all nodes in $\mathcal{A}_i^\ell \cup \mathcal{W}$ are able to decode their own messages, where the maximum value is $\lambda_{i,\ell;\mathcal{B}_i^\ell} = d_{i,\ell}$ and is reached when $\mathcal{W} = \mathcal{B}_i^\ell$; the minimum value is $\lambda_{i,\ell;\varnothing}$ and is reached when $\mathcal{W} = \varnothing$. See Fig. 3.3 for illustration.

We first take a look at the cooperation schemes with the EC hierarchy of depth 1. For the EC hierarchy of depth 1, $\mathcal{A}_i^1$ is always a subset of the neighbors of v_i, while $\mathcal{B}_i^1$ is the set of all nodes in $\mathcal{A}_j^1$, for all j such that v_j is in $\mathcal{A}_i^1$, except the ones in $\{v_i\} \cup \mathcal{A}_i^1$.

3.3.2 Single-Level Cooperation

We now discuss the case where each node only has cooperation of depth 1. Consider a DSN represented by $G(V, E)$ that is associated with parameters $(\mathbf{n}, \mathbf{k}, \mathbf{r})$ and a class of sets $\{\mathcal{M}_i\}_{v_i \in V}$ such that $\varnothing \subset \mathcal{M}_i \subseteq \mathcal{N}_i$, for all $v_i \in V$. In Construction 4, we present a joint coding scheme where node v_i only cooperates with nodes in $\mathcal{M}_i$, for all $v_i \in V$. Heterogeneity is obviously achieved since n_i, k_i, r_i, are not required to be identical for all $v_i \in V$.

Our result in previous chapter represents a special case of Construction 4, where the motivating application was in centralized cloud storage. Construction 4 extends that work to deal with arbitrary decentralized topologies, in contrast to the tree-like topology prevalent in centralized networks. Example 3 and Example 4 illustrate the efficacy of the proposed

construction in decentralized storage.

Construction 4. *Let $G(V, E)$ represent a DSN associated with parameters $(\mathbf{n}, \mathbf{k}, \mathbf{r})$ and a local EC parameter $\boldsymbol{\delta}$, where $\mathbf{r} \succ \boldsymbol{\delta} \succeq \mathbf{0}$. Let $p = |V|$ and $\mathrm{GF}(q)$ be a Galois field of size q, where $q > \max_{v_i \in V} \left(n_i + \delta_i + \sum_{v_j \in \mathcal{M}_i} \delta_j\right)$.*

For each $i \in [p]$, let $a_{i,x}$, $x \in [k_i + \delta_i]$, and $b_{i,y}$, $y \in \left[r_i + \sum_{v_j \in \mathcal{M}_i} \delta_j\right]$, be distinct elements of $\mathrm{GF}(q)$. Consider the Cauchy matrix $\mathbf{T}_i \in \mathrm{GF}(q)^{(k_i+\delta_i)\times(r_i+\sum_{v_j \in \mathcal{M}_i} \delta_j)}$ such that $\mathbf{T}_i = \mathbf{Y}(a_{i,1}, \ldots, a_{i,k_i+\delta_i}; b_{i,1}, \ldots, b_{i,r_i+\sum_{v_j \in \mathcal{M}_i} \delta_j})$. Matrix $\mathbf{G}$ in (3.1) is assembled as follows. For each $i \in [p]$, we obtain $\{\mathbf{B}_{i,j}\}_{v_j \in \mathcal{M}_i}$, $\mathbf{U}_i$, $\mathbf{A}_{i,i}$, according to the following partition of $\mathbf{T}_i$:

$$\mathbf{T}_i = \left[\begin{array}{c|c|c|c} \mathbf{A}_{i,i} & \mathbf{B}_{i,j_1} & \ldots & \mathbf{B}_{i,j_{|\mathcal{M}_i|}} \\ \hline \mathbf{U}_i & \multicolumn{3}{c}{\mathbf{Z}_i} \end{array}\right], \tag{3.3}$$

where $\mathcal{M}_i = \{v_{j_1}, v_{j_2}, \ldots, v_{j_{|\mathcal{M}_i|}}\}$, $\mathbf{A}_{i,i} \in \mathrm{GF}(q)^{k_i \times r_i}$,$\mathbf{U}_i \in \mathrm{GF}(q)^{\delta_i \times r_i}$, $\mathbf{B}_{i,j} \in \mathrm{GF}(q)^{k_i \times \delta_j}$, for $v_i \in V$ and $v_j \in \mathcal{M}_i$. Let $\mathbf{A}_{i,j} = \mathbf{B}_{i,j}\mathbf{U}_j$ if $v_j \in \mathcal{M}_i$, otherwise let it be a zero matrix.

Denote the code with generator matrix $\mathbf{G}$ by $\mathcal{C}_1$.

Theorem 2. *In a DSN with $\mathcal{C}_1$, $\mathbf{d}_i = (r_i - \delta_i, r_i + \sum_{v_j \in \mathcal{M}_i} \delta_j)$, $\mathcal{A}_i^1 = \mathcal{M}_i$, and $\mathcal{B}_i^1 = \bigcup_{v_j \in \mathcal{M}_i} (\mathcal{M}_j \setminus (\{v_i\} \cup \mathcal{M}_i))$, for all $v_i \in V$. Furthermore, the EC hierarchy associated with $d_{i,1}$ is $(\lambda_{i,1;\mathcal{W}})_{\varnothing \subseteq \mathcal{W} \subseteq \mathcal{B}_i^1}$, where*
$\lambda_{i,1;\mathcal{W}} = r_i + \sum_{j: v_j \in \mathcal{M}_i, (\mathcal{M}_j \setminus \{v_i\}) \subseteq (\mathcal{M}_i \cup \mathcal{W})} \delta_j$.

Proof. It follows directly from Lemma 3 that for all $i \in [p]$, the i-th entry of the EC hierarchy at node v_i is $\mathbf{d}_i = (r_i - \delta_i, r_i + \sum_{v_j \in \mathcal{M}_i} \delta_j)$. The remaining task is to prove that $\lambda_{i,1;\mathcal{W}} = r_i + \sum_{j: v_j \in \mathcal{M}_i, (\mathcal{M}_j \setminus \{v_i\}) \subseteq (\mathcal{M}_i \cup \mathcal{W})} \delta_j$.

For all $v_i \in V$, let $\mathbf{s}_i = \sum_{v_j \in \mathcal{M}_i} \mathbf{m}_j \mathbf{B}_{j,i}$. We first notice that for any $v_i \in V$, $v_j \in \mathcal{M}_i$, if $\mathbf{m}_j$ is recoverable, then the additional cross parities $\mathbf{s}_j \mathbf{U}_j$ and the original cross parities $\mathbf{m}_j \mathbf{A}_{j,j}$ of v_j can be computed. Therefore, $\mathbf{s}_j$ can be computed, and if all the messages $\{\mathbf{m}_{j'}\}_{v_{j'} \in \mathcal{M}_j \setminus \{v_i\}}$ are further recoverable, then the cross parities $\mathbf{m}_i \mathbf{B}_{i,j}$ of v_i from v_j can be computed from $\mathbf{m}_i \mathbf{B}_{i,j} = \mathbf{s}_j - \sum_{v_{j'} \in \mathcal{M}_j \setminus \{v_i\}} \mathbf{m}_{j'} \mathbf{B}_{j',j}$.

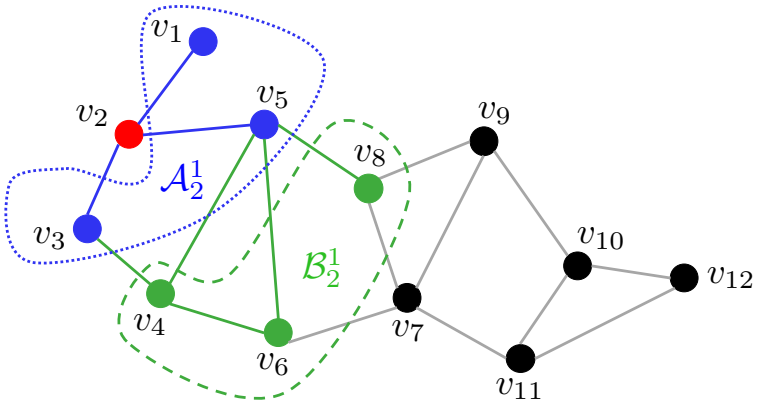


Figure 3.4: DSN for Example 3. Nodes in $\mathcal{A}_2^1$ are neighbors of v_2 and are required to be locally-recoverable to remove cross parities from the parity part of $\mathbf{c}_2$. Nodes in $\mathcal{B}_2^1$ are neighbors of nodes in $\mathcal{A}_2^1$ except for v_2 and nodes in $\mathcal{A}_2^1$ themselves. For any node v_j in $\mathcal{A}_2^1$, only when all the neighbors of v_j in $\mathcal{B}_2^1$ are recovered, v_j can provide extra parity symbols to v_2.

Previous discussion implies that for any $\mathcal{W}$, $\varnothing \subseteq \mathcal{W} \subseteq \mathcal{B}_i^1$, if $(\mathcal{M}_j \setminus \{v_i\}) \subseteq (\mathcal{M}_i \cup \mathcal{W})$, then the additional δ_j cross parities $\mathbf{m}_i\mathbf{B}_{i,j}$ of $\mathbf{m}_i$ can be obtained. Therefore, $\lambda_{i,1;\mathcal{W}} = r_i + \sum_{j:v_j\in\mathcal{M}_i,(\mathcal{M}_j\setminus\{v_i\})\subseteq(\mathcal{M}_i\cup\mathcal{W})} \delta_j$. □

1	2	3	4	5	6	7	8	9	10	11	12
$\mathbf{A}_{1,1}$	$\mathbf{B}_{1,2}\mathbf{U}_2$	0	0	0	0	0	0	0	0	0	0
$\mathbf{B}_{2,1}\mathbf{U}_1$	$\mathbf{A}_{2,2}$	$\mathbf{B}_{2,3}\mathbf{U}_3$	0	$\mathbf{B}_{2,5}\mathbf{U}_5$	0	0	0	0	0	0	0
0	$\mathbf{B}_{3,2}\mathbf{U}_2$	$\mathbf{A}_{3,3}$	$\mathbf{B}_{3,4}\mathbf{U}_4$	0	0	0	0	0	0	0	0
0	0	$\mathbf{B}_{4,3}\mathbf{U}_3$	$\mathbf{A}_{4,4}$	$\mathbf{B}_{4,5}\mathbf{U}_5$	$\mathbf{B}_{4,6}\mathbf{U}_6$	0	0	0	0	0	0
0	$\mathbf{B}_{5,2}\mathbf{U}_2$	0	$\mathbf{B}_{5,4}\mathbf{U}_4$	$\mathbf{A}_{5,5}$	$\mathbf{B}_{5,6}\mathbf{U}_6$	0	$\mathbf{B}_{5,8}\mathbf{U}_8$	0	0	0	0
0	0	0	$\mathbf{B}_{6,4}\mathbf{U}_4$	$\mathbf{B}_{6,5}\mathbf{U}_5$	$\mathbf{A}_{6,6}$	$\mathbf{B}_{6,7}\mathbf{U}_7$	0	0	0	0	0
0	0	0	0	0	$\mathbf{B}_{7,6}\mathbf{U}_6$	$\mathbf{A}_{7,7}$	$\mathbf{B}_{7,8}\mathbf{U}_8$	$\mathbf{B}_{7,9}\mathbf{U}_9$	0	$\mathbf{B}_{7,11}\mathbf{U}_{11}$	0
0	0	0	0	$\mathbf{B}_{8,5}\mathbf{U}_5$	0	$\mathbf{B}_{8,7}\mathbf{U}_7$	$\mathbf{A}_{8,8}$	$\mathbf{B}_{8,9}\mathbf{U}_9$	0	0	0
0	0	0	0	0	0	$\mathbf{B}_{9,7}\mathbf{U}_7$	$\mathbf{B}_{9,8}\mathbf{U}_8$	$\mathbf{A}_{9,9}$	$\mathbf{B}_{9,10}\mathbf{U}_{10}$	0	0
0	0	0	0	0	0	0	0	$\mathbf{B}_{10,9}\mathbf{U}_9$	$\mathbf{A}_{10,10}$	$\mathbf{B}_{10,11}\mathbf{U}_{11}$	$\mathbf{B}_{10,12}\mathbf{U}_{12}$
0	0	0	0	0	0	$\mathbf{B}_{11,7}\mathbf{U}_7$	0	0	$\mathbf{B}_{11,10}\mathbf{U}_{10}$	$\mathbf{A}_{11,11}$	$\mathbf{B}_{11,12}\mathbf{U}_{12}$
0	0	0	0	0	0	0	0	0	$\mathbf{B}_{12,10}\mathbf{U}_{10}$	$\mathbf{B}_{12,11}\mathbf{U}_{11}$	$\mathbf{A}_{12,12}$

(3.4)

Example 3. *Consider the DSN shown in Fig. 3.4. Let $\mathcal{M}_i = \mathcal{N}_i$ in Construction 4, for*

all $i \in [12]$. *The matrix in (3.4) is obtained by removing all the block columns of identity surrounded by zero matrices from the generator matrix (3.1) of* $\mathcal{C}_1$*, and is referred to as the* **non-systematic component** *of the generator matrix.*

Take node v_2 *as an example. Observe that* $\mathcal{A}_2^1 = \mathcal{M}_2 = \{v_1, v_3, v_5\}$*,* $\mathcal{A}_1^1 = \mathcal{M}_1 = \{v_2\}$*,* $\mathcal{A}_3^1 = \mathcal{M}_3 = \{v_2, v_4\}$*, and* $\mathcal{A}_5^1 = \mathcal{M}_5 = \{v_2, v_4, v_6, v_8\}$*. Therefore,* $\mathcal{B}_2^1 = \bigcup_{j\in\{1,3,5\}} \mathcal{M}_j \setminus \{v_1, v_2, v_3, v_5\} = \{v_4, v_6, v_8\}$*. Moreover,* $\mathbf{d}_2 = (r_2 - \delta_2, r_2 + \sum_{j\in\{1,3,5\}} \delta_j)$*,* $\lambda_{2,1;\varnothing} = \lambda_{2,1;\{v_6\}} = \lambda_{2,1;\{v_8\}} = \lambda_{2,1;\{v_6,v_8\}} = r_2 + \delta_1$*,* $\lambda_{2,1;\{v_4\}} = \lambda_{2,1;\{v_4,v_6\}} = \lambda_{2,1;\{v_4,v_8\}} = r_2 + \delta_1 + \delta_3$*, and* $\lambda_{2,1;\{v_4,v_6,v_8\}} = r_2 + \delta_1 + \delta_3 + \delta_5$*.*

Consider the case where the 1*-st level cooperation of* v_2 *is initiated, i.e., the number of erasures lies within the interval* $[r_2 - \delta_2 + 1, r_2 + \delta_1 + \delta_3 + \delta_5]$*. Then, if* $\mathbf{m}_1, \mathbf{m}_3, \mathbf{m}_5$ *are all locally-recoverable, the cross parities* $\mathbf{m}_1\mathbf{B}_{1,2}$*,* $\mathbf{m}_3\mathbf{B}_{3,2}$*,* $\mathbf{m}_5\mathbf{B}_{5,2}$ *computed from the non-diagonal parts in the generator matrix can be subtracted from the parity part of* $\mathbf{c}_2$ *to get* $\mathbf{m}_2\mathbf{A}_{2,2}$*. Moreover, the successful decoding of* $\mathbf{m}_1$ *makes* $\mathbf{m}_2\mathbf{B}_{2,1}$ *known to* v_2*. This process provides* $(r_2 + \delta_1)$ *parities for* $\mathbf{m}_2$*, and thus allows* v_2 *to tolerate* $(r_2 + \delta_1)$ *erasures.*

In order to correct more than $(r_2+\delta_1)$ *erasures, we need extra cross parities generated from* $\mathbf{B}_{2,3}\mathbf{U}_3$ *and* $\mathbf{B}_{2,5}\mathbf{U}_5$*. However, local decoding only allows* v_3*,* v_5 *to know* $\mathbf{m}_2\mathbf{B}_{2,3}+\mathbf{m}_4\mathbf{B}_{4,3}$ *and* $\mathbf{m}_2\mathbf{B}_{2,5} + \mathbf{m}_4\mathbf{B}_{4,5} + \mathbf{m}_6\mathbf{B}_{6,5} + \mathbf{m}_8\mathbf{B}_{8,5}$*, respectively. Therefore,* v_3 *needs* $\mathbf{m}_4$ *to be recoverable to obtain the extra* δ_3 *cross parities, and* v_5 *needs* $\mathbf{m}_4$*,* $\mathbf{m}_6$*,* $\mathbf{m}_8$ *to be recoverable to obtain the extra* δ_5 *cross parities.*

As shown in Example 3, instead of presenting a rigid EC capability, our proposed scheme enables nodes to have correction of a growing number of erasures with bigger sets of neighboring nodes recovering their messages. Therefore, nodes automatically choose the shortest path to recover their messages, thus significantly increasing the average recovery speed, especially when the erasures are distributed non-uniformly and sparsely, which is important for blockchain-based DSNs [21, 68]. Moreover, nodes with higher reliabilities are utilized to help decode the data of less reliable nodes, enabling correction of erasure patterns that are not recoverable in our previous work in [33]. We show these properties in Example 4 and

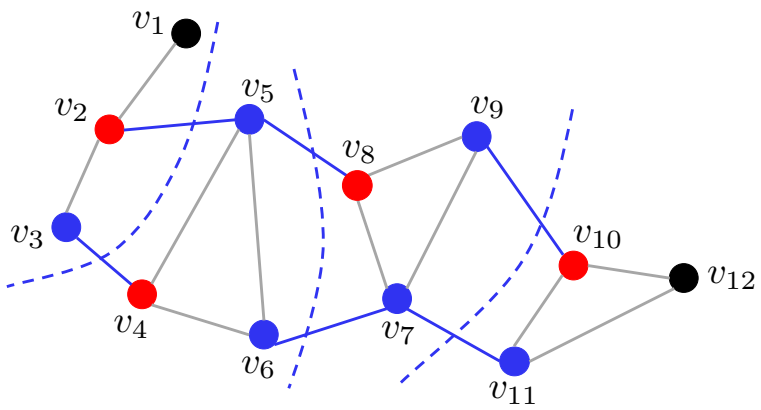


Figure 3.5: The erasure pattern in Example 5. Red and non-red nodes refer to nodes where the codewords stored at them are non-locally-recoverable and locally-recoverable, respectively.

Example 5.

Example 4. (Faster Recovery Speed) *Consider a DSN with the cooperation scheme specified in Example 3. Suppose the time to be consumed on transferring information through the communication link $e_{i,j}$ is $t_{i,j} \in \mathbb{R}^+$, where $t_{i,j} = t_{j,i}$ for all $i, j \in [12]$, $i \neq j$, and* $\max\{t_{1,2}, t_{2,5}\} < (t_{2,3} + t_{3,4}) < t_{2,5} + \min\{t_{4,5}, t_{5,6}, t_{5,8}\}$.

Consider the case where $\mathbf{c}_2$ at node v_2 has $(r_2 + 1)$ erasures, which implies that in addition to the case of $\mathbf{m}_1$, $\mathbf{m}_3$, $\mathbf{m}_5$ being obtained locally, recovering $\mathbf{m}_4$ is sufficient for v_2 to successfully obtain its message. The time consumed for decoding is $(t_{2,3} + t_{3,4})$. Therefore, any system using network coding with the property that a node failure is recovered through accessing more than 4 other nodes will need longer processing time for this case.

Example 5. (Flexible Erasure Patterns) *Consider the DSN with the cooperation scheme specified in Example 3. Suppose $\{\mathbf{m}_i\}_{i \notin \{2,4,8,10\}}$ are all locally-recoverable. Then, consider the case where $\mathbf{m}_i$ has $(r_i + 1)$ erasures for $i \in \{2, 4, 8, 10\}$, which exemplifies a correctable erasure pattern for our proposed codes.*

The hierarchical coding scheme presented in [33] can recover from this erasure pattern only if the code used adopts a partition of all nodes into 4 disjoint groups, each of which contains exactly a node from $\{v_2, v_4, v_8, v_{10}\}$, as shown in Fig. 3.5. Moreover, the partition

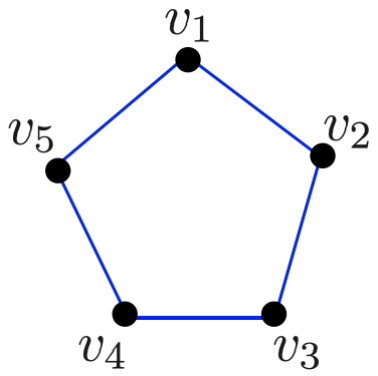


Figure 3.6: The DSN in Example 6.

of the code in [33] results in a reduction of the EC capability of the 1*-st level cooperation at every node except for* v_1, v_{12} *because the additional information Construction 4 allows to flow through the edges marked in blue no longer exists.*

Example 6. *We look at another example with DSN shown in Fig. 3.6. Suppose in our construction,* $(k_i, r_i, \delta_i) = (2, 2, 1)$*, for* $1 \leq i \leq 5$*. Then,* $d_{i,0} = r_i - \delta_i = 2 - 1 = 1$*, for all* $1 \leq i \leq 5$*. Moreover, each node* v_i *can correct* $r_i = 2$ *erasures if all its neighbor* v_{i+1}, v_{i-1} *are recoverable; it could obtain additional* 1 *parity from* v_{i+2}*,* v_{i-2}*, respectively, if they are recoverable. We next consider the total number of erasure patterns our method is able to correct. Each erasure pattern is represented by* $\mathbf{e} = (e_1, e_2, e_3, e_4, e_5) \in \mathbf{N}^5$*, and can be categorized as follows:*

1. *All nodes are locally recoverable:* $e_i \leq 1$ *for all* $1 \leq i \leq 5$*. There are* $2^5 = 32$ *such cases.*

2. *Exactly one node is non-locally recoverable:* $2 \leq e_i \leq 4$*,* $e_j \leq 1$*, for all* $j \neq i$*. There are* $5 \times 2^4 \times 3 = 240$ *such cases.*

3. *Exactly two nodes are non-locally recoverable, and all of them has at most* 3 *erasures:* $2 \leq e_i, e_{i+2} \leq 3$*,* $e_j \leq 1$*, for all* $j \neq i, i+2$*. There are* $5 \times 2^3 \times 2^2 = 160$ *such cases.*

4. *Exactly two nodes are non-locally recoverable, and one of them has exactly* 4 *erasures:*

$e_i = 4$, $2 \leq e_{i+2} \leq 3$, $e_j \leq 1$, *for all* $j \neq i, i+2$. *There are* $5 \times 2 \times 2^3 \times 2 \times 1 = 160$ *such cases.*

Therefore, there are $32 + 240 + 160 + 160 = 592$ *recoverable erasure patterns in total.*

We next consider two other schemes utilizing standard MDS codes, where there exist recoverable erasure patterns consisting of 10 *erasures in total.*

1. *A case where each node only corrects* 2 *local erasures. Note that this method achieves the fastest recovery at* 2 *local erasures, since our method can only correct* 1 *erasure locally, and need to communicate with neighboring nodes in order to correct* 2 *erasures. However, this method only corrects erasure patterns* $e_i \leq 2$, *for all* $1 \leq i \leq 5$*: there are* $3^5 = 243$ *such cases in total. Our scheme tolerates a more abundant set of erasure patterns and is able to tolerate up to* 4 *erasures at a single node with high probability.*

2. *A case where any erasure pattern with a total number of at most* 10 *erasures can be recovered. This method has the largest set of recoverable erasure patterns, and can be achieved if all the* 10 *message symbols are encoded by a* $(20, 10, 11)$ *MDS code jointly into the* 20 *encoded symbols stored among the nodes. However, by this method, even a single erasure at a node needs to access* 10 *other symbols to recover it, which can never be locally recoverable. Therefore, our scheme reaches faster recover speed than this scheme.*

We also consider the following case, where the additional cross parities of node v_i *are placed on* v_{i+2}, v_{i-2} *instead of* v_{i+1}, v_{i-1}. *Suppose communication delay on each edge is* T, *then in this case, correcting* 2 *erasures needs time* $2T$, *and correcting* $3, 4$ *erasures needs time* $3T$. *In the contrary, the original coding scheme needs time* T *for correcting* 2 *erasures and* $2T$ *for* $3, 4$ *erasures, respectively. Suppose time spent on local correction is negligible compared with the communication time* T. *Suppose each symbol is erased independently with probability* p, *where* $p < 1$. *Then, the topology-aware gain on this network is calculated as*

follows:

$$\begin{aligned}&\frac{6p^2(1-p)^2T+4p^3(1-p)(2T)+p^4(2T)}{6p^2(1-p)^2(2T)+4p^3(1-p)(3T)+p^4(3T)}\\=&\frac{6(1-p)^2+8p(1-p)+2p^2}{12(1-p)^2+12p(1-p)+3p^2}\\=&\frac{6-4p}{12-12p+3p^2}.\end{aligned}\tag{3.5}$$

When p approaches 0, the gain approaches 1/2.

In summary, our scheme is not uniformly superior to existing schemes, but achieves a customizable trade-off between the recovery speed and recoverable erasure patterns by hierarchical coding tailored for any graph with a specific topology. Note that the gain is also topology-dependent. For example, for a complete graph where each pair of nodes can communicate with identical time delay, the placement of the cross parities does not affect the time consumed on decoding. However, for a sparse planar graph, different placement of the cross parities can lead to time offsets that vary by a large scale. In future work, we expect to investigate the average topology-agnostic gain by assuming random networks under specific stochastic models such as Erdös Rényi models.

3.3.3 Recoverable Erasure Patterns

For a code specified for a DSN according to Construction 4, we next investigate the recoverable erasure patterns under the proposed EC solution. Throughout this chapter, for any edge (i, j) from v_i to v_j in a directed graph $G(V, E)$, we call v_j a child of v_i, and v_i a parent of v_j.

In the DSN depicted in Fig. 3.7, suppose all codewords stored at black nodes are locally-recoverable; those stored at green nodes, e.g., v_i with $i \in \{6, 8, 12\}$, are recoverable by accessing their neighboring nodes in $\mathcal{A}_i^1$ only; and those in blue nodes, e.g., v_i with $i \in \{0, 2, 3, 5, 10\}$, need some nodes in $\mathcal{B}_i^1$ to be also recoverable since they need to obtain extra cross parities from at least one of their neighboring nodes in $\mathcal{A}_i^1$. As an example, assume that

node v_0 needs to obtain extra parities from only one of its neighbors, say v_1. This condition requires codewords stored at $v_2, v_3, v_{13}, v_{14}, v_{15}$ all being recoverable. Since codewords in v_{13}, v_{14}, v_{15} are already locally-recoverable, this case essentially requires codewords in v_2, v_3 to be recovered. For simplicity, we just refer to this requirement as "v_0 needs v_2, v_3". Similarly, v_2 needs v_5, v_6, v_3 needs v_6, v_{10}, v_5 needs v_6, v_8, and v_{10} needs v_6, v_{12}. Given that codewords in v_6, v_8, v_{12} are recoverable, all the blue nodes are recoverable following the order v_5, v_{10}, v_2, v_3, v_0.

Note that in this chapter, we suppose that in the protocol carrying out the decoding algorithm, each node, when receiving a request, either replies back with the required message, provided that the information gathered at this node suffices to provide the answer, or broadcasts a request to all its neighbors to ask for the information it needs. For now, we assume that nodes remain intact during decoding.

We next define the so-called **decoding graph** of each node where the codeword stored there is recoverable in a DSN. For a given node, this graph describes the aforementioned order of decoding non-locally-recoverable nodes involved in the process of decoding this particular node. Observe that connections between any two nodes having their codewords locally-recoverable are omitted for simplicity.

Definition 4. (Decoding Graph) *Let $G(V, E)$ represent a DSN with $|V| = p$ and $i \in [p]$. Let $\mathcal{T}(\mathcal{V}, \mathcal{E})$ denote a directed subgraph of G associated with v_j. For all $v_i \in \mathcal{V}$, denote the set containing all children of v_i by $\mathcal{V}_i^{\mathrm{C}}$, and that containing all parents of v_i by $\mathcal{V}_i^{\mathrm{P}}$. Suppose $v_j \in \mathcal{V}$ is the only node without parents. We call this node the **root** of $\mathcal{T}$. We call any node without children a **leaf**. Suppose then the codewords of all the leaves of $\mathcal{T}$ are not locally-recoverable, and any other $v_i \in \mathcal{V}$ satisfies one of the following conditions.*

1. *The codeword stored at v_i is locally-recoverable; $\mathcal{V}_i^{\mathrm{P}} \cup \mathcal{V}_i^{\mathrm{C}}$ consists of all the nodes in $\mathcal{M}_i$ such that codewords stored at them are not locally-recoverable and $|\mathcal{V}_i^{\mathrm{P}}| = 1$.*

2. *The codeword stored at v_i is not locally-recoverable; codewords stored at nodes in $\mathcal{V}_i^{\mathrm{P}} \cup \mathcal{V}_i^{\mathrm{C}}$*

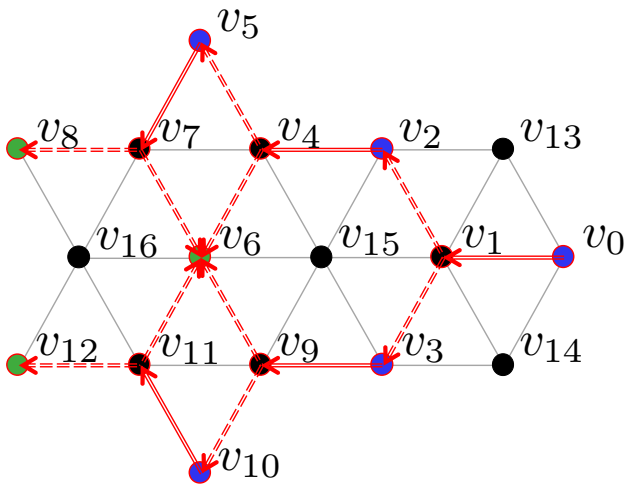


Figure 3.7: Decoding graph in Definition 4. Black and non-black nodes refer to locally-recoverable and non-locally-recoverable nodes, respectively, where green nodes are recoverable without requiring any node that is not in their neighborhood to be recovered. The subgraph marked in red is a decoding graph at root node v_0. When there exists an edge pointing from v_i to v_j, v_i needs to obtain cross parities from v_j while decoding v_0, where the edges are solid if and only if v_i is non-locally-recoverable.

are all locally-recoverable.

We call $\mathcal{T}$ a decoding graph at its root node v_j over $G(V, E)$.

As shown in Fig. 3.7, the decoding graph $\mathcal{T}$ at node v_0 is marked in red. Nodes marked in green are leaves in $\mathcal{T}$. Nodes $v_1, v_4, v_7, v_9, v_{11}$ satisfy Condition 1. Take v_1 as an example, $\mathcal{V}_1^{\mathrm{P}} = \{v_0\}$, and $\mathcal{V}_1^{\mathrm{C}} = \{v_2, v_3\}$. We know that $\mathcal{M}_1 = \{v_0, v_2, v_3, v_{13}, v_{14}, v_{15}\}$, where codewords stored at nodes in $\mathcal{V}_i^{\mathrm{P}} \cup \mathcal{V}_i^{\mathrm{C}} = \{v_0, v_2, v_3\}$ are not locally-recoverable. This local constraint enforces the node in $\mathcal{V}_1^{\mathrm{P}}$, i.e., v_0, to obtain the extra cross parities from v_1 after codewords stored at nodes in $\mathcal{V}_1^{\mathrm{C}}$ are recovered. Nodes $v_0, v_2, v_3, v_5, v_6, v_{10}, v_8, v_{12}$ satisfy Condition 2, and they are the nodes that need to recover their codewords in order that v_0 recovers its codeword.

Based on the definition of decoding graphs, Theorem 3 describes recoverable erasure patterns in a DSN.

Theorem 3. *Let $\mathcal{C}$ be a code with single-level cooperation on a DSN represented by $G(V, E)$, where $\mathcal{C}$ and all related parameters are specified according to Construction 4. Let* $\mathbf{u} \in \mathbb{N}^p$

such that $\mathbf{u} \preceq \mathbf{n}$. *Suppose* $\mathcal{C}$ *and* $\mathbf{u}$ *satisfy the following conditions:*

1. *Let* V^{NL} *represent the set that contains all the nodes* v_i, $i \in [p]$ *such that* $u_i > r_i - \delta_i$. *Let* $V^{\mathrm{L}} = V \setminus V^{\mathrm{NL}}$. *Then, for any* $v_i \in V^{\mathrm{NL}}$, $\mathcal{M}_i \subset V^{\mathrm{L}}$.

2. *For any* $v_i \in V^{\mathrm{NL}}$, *there exists a decoding graph* $\mathcal{T}_i(\mathcal{V}_i, \mathcal{E}_i)$ *at root* v_i *over* G. *Moreover, for any leaf* v_j *of* $\mathcal{T}_i$, $u_j \leq r_j$; *for any node* $v_j \in \mathcal{V}_i \cap V^{\mathrm{NL}}$, $u_j \leq r_j + \sum_{v_k \in \mathcal{V}_j^{\mathrm{C}}} \delta_k$.

Then, $\mathbf{u}$ *is a* **recoverable erasure pattern** *of* $\mathcal{C}$ *over* $G(V, E)$.

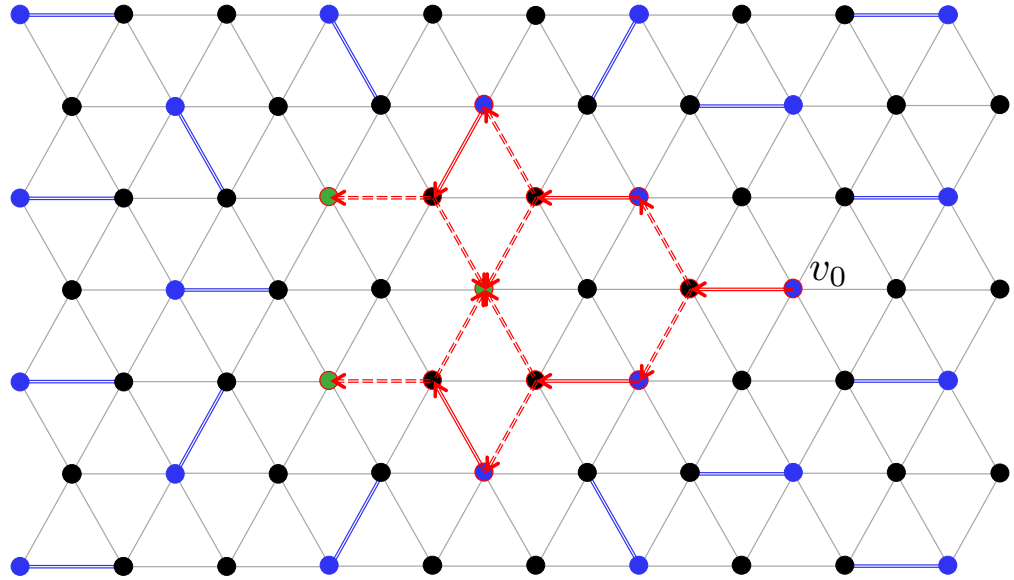
Proof. For any node $v_i \in \mathcal{V}^{\mathrm{NL}}$, consider the decoding graph $\mathcal{T}_i(\mathcal{V}_i, \mathcal{E}_i)$ at root v_i. Denote the number of nodes contained in V^{L} on the longest directed path connecting node v_i with a leaf in $\mathcal{T}_i$ by l_i, which is referred to as the **decoding depth** of v_i. We prove the statement "any node in G is recoverable" by mathematical induction on the decoding depth of the node.

The decoding graph of a node with decoding depth 0 contains only the node itself. The first condition implies that all neighbors of any node that is not locally-recoverable (in $\mathcal{V}^{\mathrm{NL}}$) are locally-recoverable (in $\mathcal{V}^{\mathrm{L}}$). Therefore, any node $v_i \in \mathcal{V}^{\mathrm{NL}}$ tolerates at least r_i erasures. This means that nodes with decoding depth 0 are recoverable.

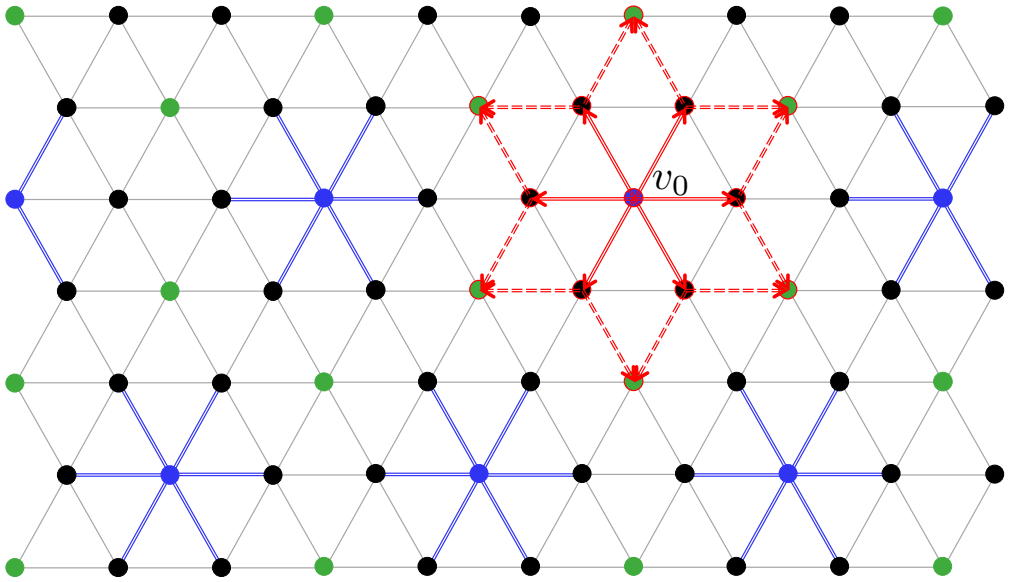
Suppose the statement is true for any node $v_j \in \mathcal{V}^{\mathrm{NL}}$ with decoding depth less than or equal to $\ell \in \mathbb{N}$. Then, for any node v_i with decoding depth $\ell + 1$, let S_i denote the union of all sets $\mathcal{V}_j^{\mathrm{C}}$ such that $v_j \in \mathcal{V}_i^{\mathrm{C}}$. Since the subgraph of $\mathcal{T}_i$ rooted at any node $v_j \in S_i$ is a decoding graph of v_j with length at most ℓ, v_j is recoverable. Condition 1) in Definition 4 indicates that all neighbors of $v_j \in \mathcal{V}_i^{\mathrm{C}}$ except for v_i are recoverable and $\mathbf{m}_i \mathbf{B}_{i,j}$ is known, which provides δ_j extra parities of $\mathbf{m}_i$. Therefore, node v_i tolerates up to $r_i + \sum_{v_j \in \mathcal{V}_i^{\mathrm{C}}} \delta_j$ erasures, thus is recoverable according to Condition 2). Consequently, the statement for $\ell + 1$ is also true.

By induction, the statement is true for all the nodes, and the theorem is true. □

The following two examples illustrate Theorem 3. They follow the notation in Construction 4 and Theorem 3.



(a)



(b)

Figure 3.8: DSN in Example 7. Black and non-black nodes refer to nodes that are locally-recoverable and non-locally-recoverable, respectively. The decoding graphs at v_0 are marked with color red. Any solid blue line connects a black node v_i and an non-black node v_j, where v_i is a child of v_j, and all neighbors of v_i except for v_j are children of v_i, in the decoding graph at root v_j. The blue lines uniquely describe the decoding graphs at each non-black node.

Example 7. *Fig. 3.8 presents two erasure patterns,* $\mathbf{u}_1 = (u_{1,1}, u_{1,2}, \ldots, u_{1,p})$ *(left) and* $\mathbf{u}_2 = (u_{2,1}, u_{2,2}, \ldots, u_{2,p})$ *(right), on the same DSN denoted by* $G(V, E)$ *with the EC solution characterized by* $\mathcal{C}$ *specified in Construction 4. Suppose there exists* $\delta \in \mathbb{N}$ *such that* $\delta_i = \delta$ *for all* $i \in [p]$.

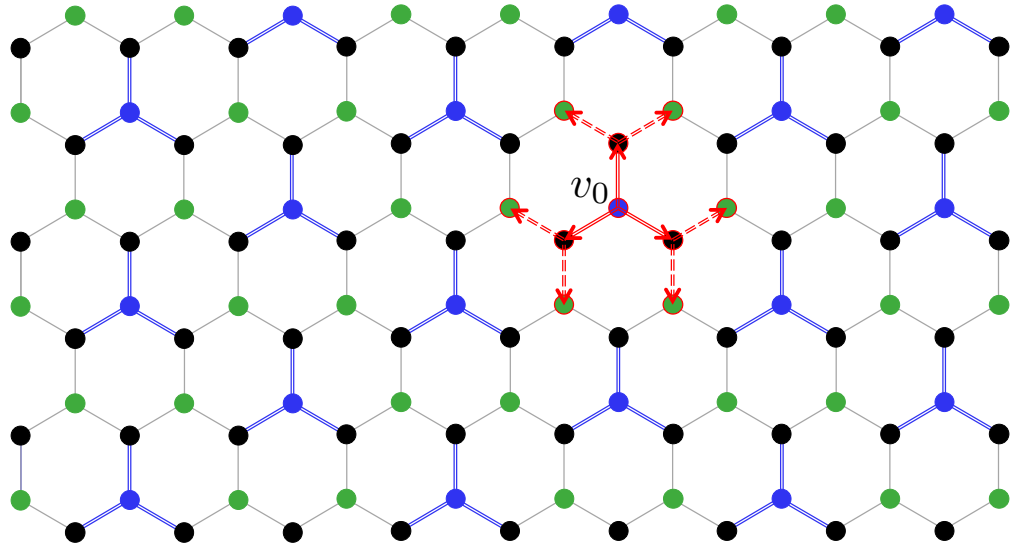
For any $i \in [p]$, v_i *is marked in black if* $u_{j,i} \leq r_i - \delta$, *in green if* $r_i - \delta < u_{j,i} \leq r_i$, *and in blue otherwise, where* $j \in \{0, 1\}$. *In the left panel, that specifies* $\mathbf{u}_1$, *any node* v_i *marked in blue satisfies* $r_i < u_{1,i} \leq r_i + \delta$. *In the right panel, that specifies* $\mathbf{u}_2$, *any node* v_i *marked in blue satisfies* $r_i < u_{2,i} \leq r_i + 6\delta$.

Note that any non-black node is connected to exactly one black node by a blue edge, where the non-black node is the only parent of the black node in Definition 4. Then, for any node v_i *in* $G(V, E)$, *there exists a decoding graph at* v_i, *with the leaves being all marked in green. In Fig. 3.8, the decoding graph at the node* v_0 *is marked in red on each graph of the two. Let* d_i *be the maximum number of erasures node* v_i *tolerates, for all* $i \in [p]$, *and* $\Delta_i = d_i - r_i$. *Suppose* $p \to \infty$ *in* $G(V, E)$. *Denote the average of all* Δ_i*'s by* Δ.

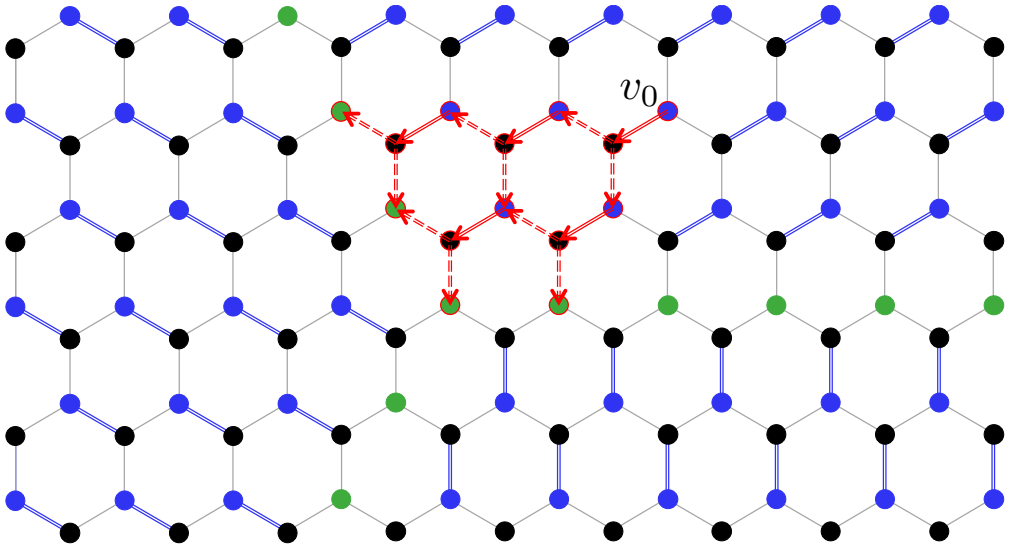
In the first subgraph, there will be approximately $2p/3$ *nodes with any* v_i *of them satisfying* $\Delta_i = -\delta$, *and approximately* $p/3$ *nodes with any* v_i *of them satisfying* $\Delta_i = \delta$. *Thus,* $\Delta = -\delta/3$. *Similarly, in the second graph, there will be approximately* $2p/3$, $2p/9$, *and* $p/9$ *nodes with any* v_i *of them satisfying* $\Delta_i = -\delta$, $\Delta_i = 0$, *and* $\Delta_i = 6\delta$, *respectively. Thus,* $\Delta = 0$.

Example 8. *Similar to Example 7, Fig. 3.9 also presents two erasure patterns on another DSN with an EC solution characterized by* $\mathcal{C}$ *specified in Construction 4. Suppose there exists* $\delta \in \mathbb{N}$ *such that* $\delta_i = \delta$ *for all* $i \in [p]$.

For any $i \in [p]$, v_i *is marked in black if* $u_{j,i} \leq r_i - \delta$, *in green if* $r_i - \delta < u_{j,i} \leq r_i$, *and in blue otherwise, where* $j \in \{0, 1\}$. *In the left panel, that specifies* $\mathbf{u}_1$, *any node* v_i *marked in blue satisfies* $r_i < u_{1,i} \leq r_i + 3\delta$. *In the right panel, that specifies* $\mathbf{u}_2$, *any node* v_i *marked in blue satisfies* $r_i < u_{2,i} \leq r_i + \delta$. *In Fig. 3.9, the decoding graph at the node* v_0 *is marked in red on both graphs of the two.*



(a)



(b)

Figure 3.9: DSN in Example 8. Meanings of the components of the graphs are identical to those in Fig. 3.8.

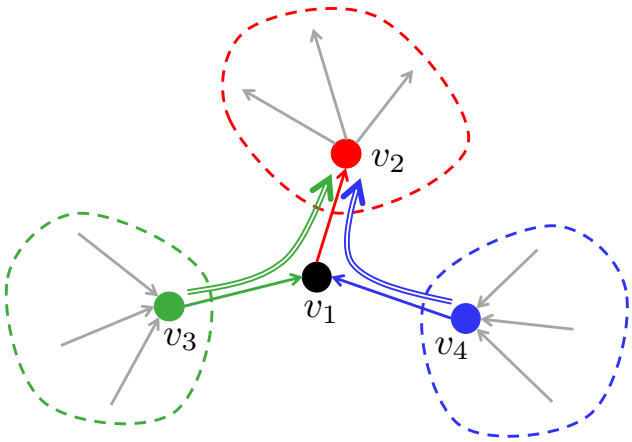


Figure 3.10: Information flow in cooperative data protection. As a neighbor of v_2, v_1 helps in removing the cross parities from the parity part of $\mathbf{c}_2$. However, nodes v_3 and v_4 also indirectly help v_1 to provide extra cross parities to v_2 if they are recovered. This can be interpreted as information flow from v_3 and v_4 to v_2 through v_1.

We follow the definitions of Δ_i's, $i \in [p]$, and Δ, stated in Example 7. In the left panel, there will be approximately $p/2$ nodes with any v_i of them satisfying $\Delta_i = -\delta$, approximately $p/3$ nodes with any v_i of them satisfying $\Delta_i = 0$, and $p/6$ nodes with any v_i of them satisfying $\Delta_i = 3\delta$. Thus, $\Delta = 0$. Similarly, in the right panel, there will be approximately $p/2$ and $p/2$ nodes with any v_i of them satisfying $\Delta_i = -\delta$ and $\Delta_i = \delta$, respectively. Thus, $\Delta = 0$.

Remark 2. (Information Flow in Coded DSN) *Note that the values of Δ_i's, $i \in [p]$, in Example 7 and Example 8 imply the unbalanced reliabilities of nodes in the coded DSN. In other words, any node (black) v_i with $\Delta_i < 0$ is of higher reliability than any node (blue) v_i with $\Delta_i > 0$. Therefore, any blue node utilizes extra information from non-black nodes in its neighborhood. The average Δ being nonnegative can be interpreted as a higher level of intrinsic information flow among nodes with different reliabilities in the coded DSN.*

For example, as shown in Fig. 3.10, suppose v_1 is a locally-recoverable node with neighboring nodes v_2, v_3, and v_4. If v_3 and v_4 are recoverable, then v_1 provides δ_1 extra parities to node v_2. Therefore, the information flows from v_3 and v_4 to v_2 through v_1, which is depicted in the figure.

3.4 Multi-Level Cooperation

In this section, we extend the construction presented in Subsection 3.3.2 to codes with EC hierarchies of depth larger than 1. As is shown in schemes with single-level cooperation, cooperation utilizes the redundant information from nodes with fewer erasures to help in decoding of nodes that cannot be decoded locally. However, each node only obtains additional parities from its neighbors in the single-level cooperation, which immediately motivates us to explore multi-level cooperation to further improve the global EC capability of each node. Although multi-level cooperation inevitably degrades the local EC capability of each node, it enables the DSN to tolerate erasure patterns where erasures are distributed non-uniformly among the nodes, such as bursty erasures in few sparsely scattered nodes. In this section, we investigate the EC hierarchy of multi-level cooperation schemes. We first define the so-called **cooperation graphs** that describe how the nodes are coupled to cooperatively transmit information, and then prove the existence of hierarchical codes over a special class of cooperation graphs: the so-called **compatible graphs**.

1	2	3	4	5	6	7	8	9	10	11	12
$\mathbf{A}_{1,1}$	$\mathbf{B}_{1,2}\mathbf{U}_2$	0	0	0	0	0	0	0	0	0	0
$\mathbf{B}_{2,1}\mathbf{U}_1$	$\mathbf{A}_{2,2}$	$\mathbf{B}_{2,3}\mathbf{U}_3$	0	$\mathbf{B}_{2,5}\mathbf{U}_5$	0	0	$\mathbf{B}_c\mathbf{V}_{8;2}$	$\mathbf{B}_c\mathbf{V}_{9;2}$	$\mathbf{B}_d\mathbf{V}_{10;3}$	$\mathbf{B}_d\mathbf{V}_{11;3}$	0
0	$\mathbf{B}_{3,2}\mathbf{U}_2$	$\mathbf{A}_{3,3}$	$\mathbf{B}_{3,4}\mathbf{U}_4$	0	0	0	$\mathbf{B}_e\mathbf{V}_{8;2}$	$\mathbf{B}_e\mathbf{V}_{9;2}$	$\mathbf{B}_f\mathbf{V}_{10;3}$	$\mathbf{B}_f\mathbf{V}_{11;3}$	0
0	0	$\mathbf{B}_{4,3}\mathbf{U}_3$	$\mathbf{A}_{4,4}$	$\mathbf{B}_{4,5}\mathbf{U}_5$	$\mathbf{B}_{4,6}\mathbf{U}_6$	0	$\mathbf{B}_\alpha\mathbf{V}_{8;2}$	$\mathbf{B}_\alpha\mathbf{V}_{9;2}$	$\mathbf{B}_g\mathbf{V}_{10;2}$	$\mathbf{B}_g\mathbf{V}_{11;2}$	0
0	$\mathbf{B}_{5,2}\mathbf{U}_2$	0	$\mathbf{B}_{5,4}\mathbf{U}_4$	$\mathbf{A}_{5,5}$	$\mathbf{B}_{5,6}\mathbf{U}_6$	0	$\mathbf{B}_{5,8}\mathbf{U}_8$	0	0	$\mathbf{B}_h\mathbf{V}_{11;2}$	$\mathbf{B}_h\mathbf{V}_{12;2}$
0	0	0	$\mathbf{B}_{6,4}\mathbf{U}_4$	$\mathbf{B}_{6,5}\mathbf{U}_5$	$\mathbf{A}_{6,6}$	$\mathbf{B}_{6,7}\mathbf{U}_7$	$\mathbf{B}_\beta\mathbf{V}_{8;2}$	$\mathbf{B}_\beta\mathbf{V}_{9;2}$	$\mathbf{B}_j\mathbf{V}_{10;2}$	0	$\mathbf{B}_j\mathbf{V}_{12;2}$
0	0	0	0	0	$\mathbf{B}_{7,6}\mathbf{U}_6$	$\mathbf{A}_{7,7}$	$\mathbf{B}_{7,8}\mathbf{U}_8$	$\mathbf{B}_{7,9}\mathbf{U}_9$	$\mathbf{B}_l\mathbf{V}_{10;2}$	$\mathbf{B}_{7,11}\mathbf{U}_{11}$	$\mathbf{B}_l\mathbf{V}_{12;2}$
0	$\mathbf{B}_m\mathbf{V}_{2;2}$	$\mathbf{B}_m\mathbf{V}_{3;2}$	$\mathbf{B}_y\mathbf{V}_{4;2}$	$\mathbf{B}_{8,5}\mathbf{U}_5$	$\mathbf{B}_y\mathbf{V}_{6;2}$	$\mathbf{B}_{8,7}\mathbf{U}_7$	$\mathbf{A}_{8,8}$	$\mathbf{B}_{8,9}\mathbf{U}_9$	$\mathbf{B}_n\mathbf{V}_{10;2}$	$\mathbf{B}_n\mathbf{V}_{11;2}$	0
0	$\mathbf{B}_o\mathbf{V}_{2;2}$	$\mathbf{B}_o\mathbf{V}_{3;2}$	$\mathbf{B}_z\mathbf{V}_{4;2}$	0	$\mathbf{B}_z\mathbf{V}_{6;2}$	$\mathbf{B}_{9,7}\mathbf{U}_7$	$\mathbf{B}_{9,8}\mathbf{U}_8$	$\mathbf{A}_{9,9}$	$\mathbf{B}_{9,10}\mathbf{U}_{10}$	$\mathbf{B}_p\mathbf{V}_{11;2}$	$\mathbf{B}_p\mathbf{V}_{12;2}$
0	$\mathbf{B}_q\mathbf{V}_{2;3}$	$\mathbf{B}_q\mathbf{V}_{3;3}$	$\mathbf{B}_r\mathbf{V}_{4;2}$	0	$\mathbf{B}_r\mathbf{V}_{6;3}$	$\mathbf{B}_s\mathbf{V}_{7;3}$	$\mathbf{B}_s\mathbf{V}_{8;3}$	$\mathbf{B}_{10,9}\mathbf{U}_9$	$\mathbf{A}_{10,10}$	$\mathbf{B}_{10,11}\mathbf{U}_{11}$	$\mathbf{B}_{10,12}\mathbf{U}_{12}$
0	$\mathbf{B}_x\mathbf{V}_{2;3}$	$\mathbf{B}_x\mathbf{V}_{3;3}$	$\mathbf{B}_t\mathbf{V}_{4;2}$	$\mathbf{B}_t\mathbf{V}_{5;2}$	0	$\mathbf{B}_{11,7}\mathbf{U}_7$	$\mathbf{B}_u\mathbf{V}_{8;3}$	$\mathbf{B}_u\mathbf{V}_{9;3}$	$\mathbf{B}_{11,10}\mathbf{U}_{10}$	$\mathbf{A}_{11,11}$	$\mathbf{B}_{11,12}\mathbf{U}_{12}$
0	0	0	0	$\mathbf{B}_v\mathbf{V}_{5;2}$	$\mathbf{B}_v\mathbf{V}_{6;3}$	$\mathbf{B}_w\mathbf{V}_{7;3}$	0	$\mathbf{B}_w\mathbf{V}_{9;3}$	$\mathbf{B}_{12,10}\mathbf{U}_{10}$	$\mathbf{B}_{12,11}\mathbf{U}_{11}$	$\mathbf{A}_{12,12}$

(3.6)

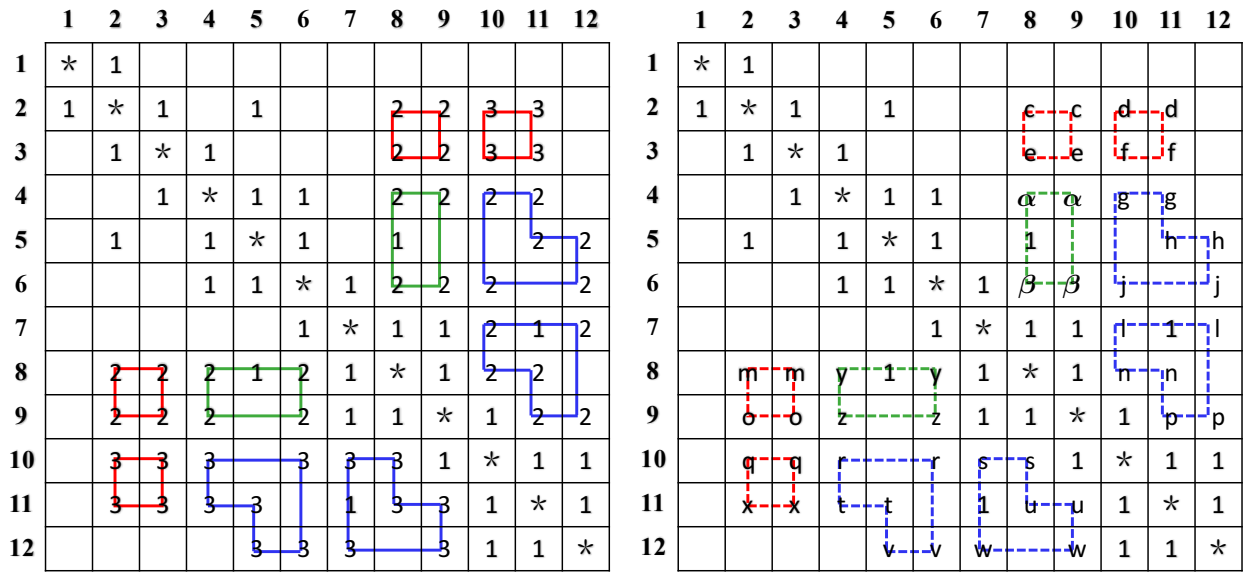


Figure 3.11: Matrices $\mathbf{D}$ (left) and $\mathbf{X}$ (right) in Example 9. A numerical entry ℓ at position (i, j) in the left panel implies that v_j is adjacent to v_i in the ℓ-th level cooperation of v_i, while symbolic entries (letters) in the right panel represent the indices of the component matrices $\mathbf{A}_{i,j}$.

3.4.1 Cooperation Graphs

Based on the aforementioned notation, for each $v_i \in V$ and $\ell \in [L_i]$, let $\mathcal{I}_i^\ell = \mathcal{A}_i^\ell \setminus \mathcal{A}_i^{\ell-1}$ (with $\mathcal{A}_i^0 = \varnothing$) and refer to it as the ℓ-th **helper** of v_i. We next define the so-called **cooperation matrix**.

Definition 5. *For a joint coding scheme $\mathcal{C}$ for a DSN represented by $G(V, E)$ with $|V| = p$, the matrix $\mathbf{D} \in \mathbb{N}^{p\times p}$, in which $\mathbf{D}_{i,j}$ equals to ℓ for all $i, j \in [p]$ such that $j \in \mathcal{I}_i^\ell$, $\ell \in [L_i]$, and zero otherwise, is called the* ***cooperation matrix****.*

As an example, the cooperation matrix in Example 3 is exactly the adjacency matrix of the graph in Fig. 3.4. Note that cooperation graphs corresponding to some joint coding schemes must satisfy certain properties. In Subsection 3.4.2, we prove the existence of codes if the cooperation matrix represents a so-called **compatible graph**. Before going into details of the construction, we present an example to provide some intuition.

Example 9. *Recall the DSN in Example 3. We present a coding scheme with the cooperation matrix specified in the left panel of Fig. 3.11. The non-systematic part of the generator matrix is shown in (3.6), which is obtained through the following process:*

1. *Partition all the non-zero, non-one elements into structured groups, each of which is marked in either a rectangle or a hexagon in* $\mathbf{D}$*, as indicated in the left panel of Fig. 3.11.*

2. *Replace the endpoints of each horizontal line segment in Step 1 with* $s \in S$ *(*S *is a set of symbols), as indicated in the right panel of Fig. 3.11; denote the new matrix by* $\mathbf{X}$*.*

3. *Assign a parameter* $\gamma_s \in \mathbb{N}$ *to each* $s \in S$*, and a matrix* $\mathbf{B}'_s \in \mathrm{GF}^{k_i \times \gamma_s}$ *to any* (i, j) *such that* $\mathbf{X}_{i,j} = s$*.*

4. *For each* $i \in [p]$*,* $\ell \in [L]$*, let* $\eta_{i;\ell} = \max_{s:k\in\mathcal{I}_i^\ell, \mathbf{X}_{k,i}=s} \gamma_s$*, assign* $\mathbf{V}_{i;\ell} \in \mathrm{GF}(q)^{\eta_{i;\ell}\times r_i}$ *to* v_i*; let* $\mathbf{B}_s = \left[\mathbf{B}'_s, \mathbf{0}_{\eta_{i;\ell}-\gamma_s}\right]$*; compute* $\mathbf{A}_{i,j} = \mathbf{B}_s\mathbf{V}_{j;\ell}$ *for* $s = \mathbf{X}_{i,j}$*,* $l = \mathbf{D}_{i,j}$*.*

5. *Compute* $\mathbf{A}_{i,j}$ *for* $\mathbf{X}_{i,j} = 1$ *according to Construction 4.*

Note that the colors of submatrices in (3.6) are consistent with the colors of cycles in Fig. 3.11. Let us again focus on node v_2*. Let* $\mathcal{I}_2^1 = \{v_1, v_3, v_5\}$*,* $\mathcal{I}_2^2 = \{v_8, v_9\}$*,* $\mathcal{I}_2^3 = \{v_{10}, v_{11}\}$*. Then,* $\mathcal{B}_2^1 = \{v_4, v_6, v_8\}$*,* $\mathcal{B}_2^2 = \{v_4, v_6\}$*,* $\mathcal{B}_2^3 = \varnothing$*,* $d_{2,0} = r_2-\delta_2-\eta_{i;2}-\eta_{i;3}$*,* $d_{2,1} = r_2+\delta_1+\delta_3+\delta_5$*,* $d_{2,2} = d_{2,1}+\gamma_c$*,* $d_{2,3} = d_{2,2}+\gamma_d$*. Note that for each* $i \in [p]$*,* $s = \mathbf{X}_{i,j}$*, and* $l = \mathbf{D}_{i,j}$*,* γ_s *denotes the maximum number of parity symbols* v_i *can obtain from* v_j *in the* ℓ*-th level cooperation, and* $\eta_{i;\ell}$ *represents the reduction in the value of the local erasure correction capability needed at* v_i *because of its* ℓ*-th level cooperation.*

We first show that knowing $\{\mathbf{m}_j\}_{v_j\in\mathcal{A}_2^1}$ *is sufficient for removing* $\mathbf{s}_2 = \sum_{j\in\mathcal{I}_2^1} \mathbf{m}_j\mathbf{B}_{j,2}\mathbf{U}_2 + \sum_{\ell=2}^{3}\sum_{j\in\mathcal{I}_2^\ell} \mathbf{m}_j\mathbf{B}_{\mathbf{X}_{j,2}}\mathbf{V}_{2;\ell}$ *from the parity part of* $\mathbf{c}_2$*. Note that if the rows of* $\mathbf{A}_{i,i}$*,* $\mathbf{U}_i$*, and* $\{\mathbf{V}_{i;\ell}\}_{\ell\in\{2,3\}}$ *are linearly independent, then for all* ℓ*,* $\sum_{j\in\mathcal{I}_i^\ell} \mathbf{m}_j\mathbf{B}_{\mathbf{X}_{j,i}}$ *is recoverable if* $\mathbf{m}_i$ *is recoverable. In our example, this means that* $\{\mathbf{m}_j\mathbf{B}_{j,2}\}_{j=1,3,5}$*,* $\mathbf{m}_8\mathbf{B}_m+\mathbf{m}_9\mathbf{B}_o$*,* $\mathbf{m}_{10}\mathbf{B}_o+\mathbf{m}_{11}\mathbf{B}_x$ *are known, which means* $\mathbf{s}_2$ *is also known. Therefore,* $\mathbf{s}_2$ *is removed from the parity part of* $\mathbf{c}_2$

through the 1*-st level cooperation. We next show that additional parities are obtained through* ℓ*-th level cooperations with* $\ell \in \{2, 3\}$.

In the 2*-nd level cooperation,* $\mathbf{m}_8, \mathbf{m}_9$ *are known. Therefore,* $\mathbf{m}_2\mathbf{B}_c + \mathbf{m}_3\mathbf{B}_e + \mathbf{m}_4\mathbf{B}_\alpha + \mathbf{m}_6\mathbf{B}_\beta$ *is also known. We remove* $\mathbf{m}_3\mathbf{B}_e$*, that is obtained via* v_3*, from the parity part of* $\mathbf{c}_2$*. In order to obtain the* γ_c *parities from* $\mathbf{m}_2\mathbf{B}_c$*, one needs* $\mathbf{m}_4, \mathbf{m}_6$ *to be recoverable. Therefore,* $\mathcal{B}_2^2 = \{v_4, v_6\}$, $d_{2,2} = d_{2,1} + \gamma_c$, $\lambda_{2,2;\varnothing} = d_{2,1}$.

The cooperation matrix adopts a partition of non-zero, non-one elements into groups where each of them forms a cycle (see Example 9). Suppose there are T cycles. Represent each cycle with index $t \in [T]$ by a tuple $C_t = (X_t, Y_t, \{X_{t;j}\}_{j\in Y_t}, \{Y_{t;i}\}_{i\in X_t}, g_t, (\ell_{t;j})_{j\in Y_t})$, where X_t and Y_t denote the sets containing indices of the rows and the columns of the cycle, respectively. Let $X_{t;j} = \{i_1, i_2\}$ for $j \in Y_t$, where $(i_1, j), (i_2, j)$ are the vertices of the cycle C_t with column index j. Let $Y_{t;i} = \{j_1, j_2\}$ for $i \in X_t$, where $(i, j_1), (i, j_2)$ are the vertices of the cycle C_t with row index i. Let g_t denote a group number assigned to the cycle C_t, which will be explained shortly. Observe that any two vertices of a cycle that share the same column have the same cooperation level. Let $\ell_{t;j}$ denote the number representing the cooperation level assigned to the vertices $(i_1, j), (i_2, j)$ of the cycle C_t where $X_{t;j} = \{i_1, i_2\}$. Suppose values in $(g_t)_{t\in[T]}$ span all the values in $[A]$, for some $A \in \mathbb{N}$. For any $g \in [A]$, denote the set containing all t such that $g_t = g$ by T_g.

For example, let $t = 1$ for the blue cycle at the bottom left panel of the matrices in Fig. 3.11. Then, the cycle C_1 is represented by $(\{10, 11, 12\}, \{4, 5, 6\}, \{X_{1;j}\}_{j=4}^{6}, \{Y_{1;i}\}_{i=10}^{12}, 1, (\ell_{1;j})_{j=4}^{6})$, where $X_{1;4} = \{10, 11\}$, $X_{1;5} = \{11, 12\}$, $X_{1;6} = \{10, 12\}$, $Y_{1;10} = \{4, 6\}$, $Y_{1;11} = \{4, 5\}$, $Y_{1;12} = \{5, 6\}$, and $\ell_{1;4} = \ell_{1;5} = \ell_{1;6} = 3$.

Observe that cycle C_t, $t \in [T]$, in Fig. 3.11 essentially represents a cycle in the complementary graph[1] $\bar{G}$ of G on V, since each vertex (i, j) on the cycle implies that $\mathbf{D}_{i,j} \neq 1$, i.e., there is no edge connecting v_i and v_j in G, and there is an edge (i, j) in the complementary graph $\bar{G}$. Cycle C_t can also be interpreted as a pair of non-adjacent edges or non-overlapping

[1]The complementary graph of a graph $G(V, E)$ consists of all nodes in V and all edges that are not in E.

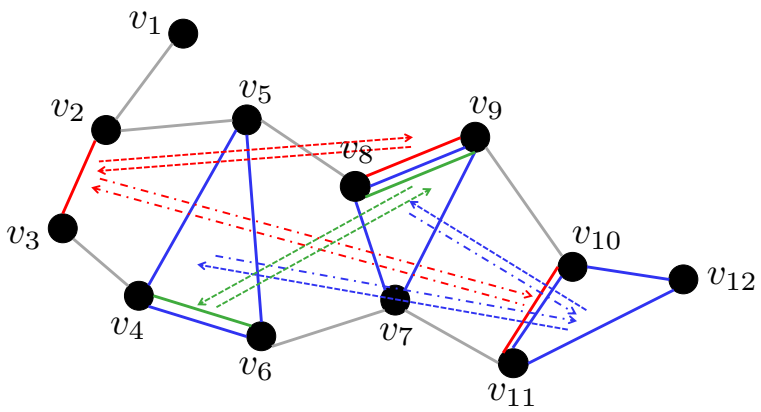


Figure 3.12: Cooperation graph of Example 9. Dashed directed edges are used for cooperation level 2, and dash-dotted directed edges are used for cooperation level 3.

triangles in G with vertices from X_t and Y_t. We mark X_t, Y_t, add an edge with arrow and the label g_t that points from X_t to Y_t for each $t \in [T]$, and obtain the so-called **cooperation graph**[2]. The cooperation graph for the coding scheme in Example 9 is shown in Fig. 3.12.

From the aforementioned description, a cooperation graph does not necessarily lead to a unique cooperation matrix, since the latter requires not only to further specify the associated set of cooperation levels, but also to identify the associated local matching graph, to be defined soon, for each cycle C_t if X_t has more than two nodes. For example, Fig. 3.13 presents three out of a total of six possible ways to specify $\{X_{t;j}\}_{j\in Y_t}$ and $\{Y_{t;i}\}_{i\in X_t}$ for a cooperation cycle C_t with $X_t = \{i_1, i_2, i_3\}$ and $Y_t = \{j_1, j_2, j_3\}$. As indicated by Fig. 3.13, each specified cycle of the three is uniquely represented by the set $\{(i,j)\}_{i\in X_t, Y_t\setminus Y_{t;i}=\{j\}}$ of edges that are not on the cycle. In these three graphs, if a double-dashed blue line with bidirectional arrows connects node v_{i_x} to node v_{j_y}, this means that v_{i_x} cooperates with the two nodes in $Y_t \setminus \{v_{j_y}\}$ in the cooperation involving vertices on the two triangles containing v_{j_x}, v_{i_y}, respectively. We refer to the resulting graph as a **matching graph** corresponding to a cycle. If there is more than one cycle involved in the matching graph, we call that graph

[2]Note that there are two types of edges in the graph. The edges of one type connect nodes that are directly connected in their 1-st level cooperation, and these are solid edges in the graph. The edges of the other type point between two groups of nodes (edges or triangles) that are adjacent in their higher level cooperation, and these are dashed arrows in the graph. For simplicity, "connected" here is used with the meaning of "adjacent".

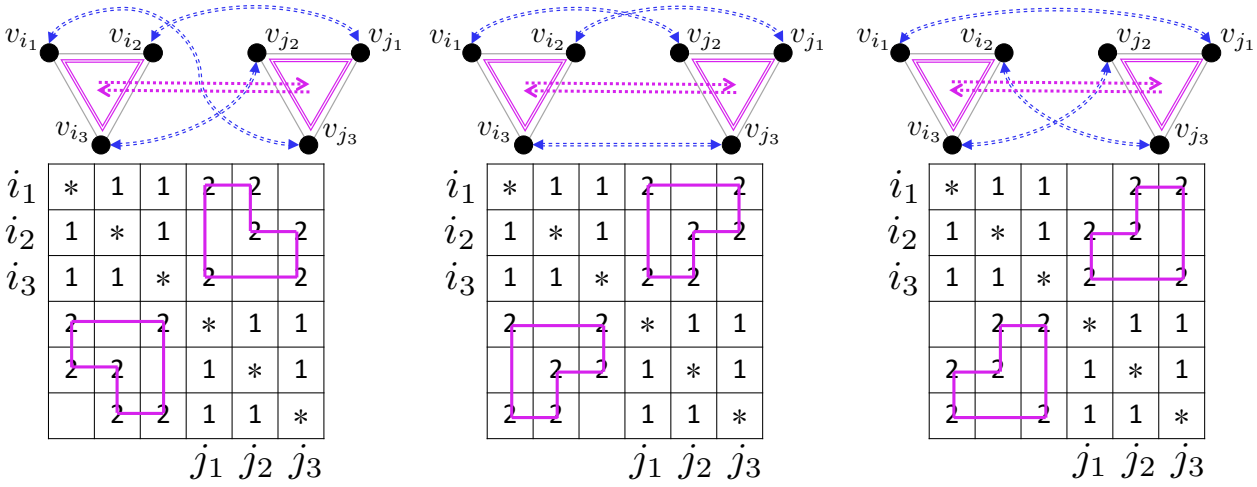


Figure 3.13: Possible local matching graphs and their corresponding local cooperation matrices contained in a multi-level cooperation graph between 6 nodes.

a **local matching graph**. If cycles in the local matching graph are all the cycles of a cycle group, we call that graph an **isolated local matching graph** and will discuss it in detail in Subsection 3.4.2.

Observe that although in Example 9, the cooperation levels $(\ell_{t;j})_{t\in[T],j\in Y_t}$ specified for all the nodes on any cycle C_t, $t \in [T]$, are identical, this case is not a necessary condition. We present an example in Example 10, in which cooperation levels for nodes on the same cycle can be different. This example provides intuition both in deciding conditions that ensure a graph to be a cooperation graph, and in how to assign cooperation levels to such a cooperation graph if it exists.

In Subsection 3.4.2, we introduce the method of assigning cooperation levels over a given cooperation graph to obtain a so-called **compatible graph**. The algorithm to find a cooperation graph over a DSN $G(V, E)$ with a given topology is described in Subsection 3.5.1.

Example 10. *The left panel of Fig. 3.14 presents the cycle representation of a subgraph of a compatible graph on a DSN. Denote nodes associated with the left-most column to the right-most column by v_1 to v_{28} in order, and let Fig. 3.15 represent the subgraph containing nodes $\{v_i\}_{5\le i\le 22}$ of the cooperation graph. The right panel of Fig. 3.14 denotes the cooperation*

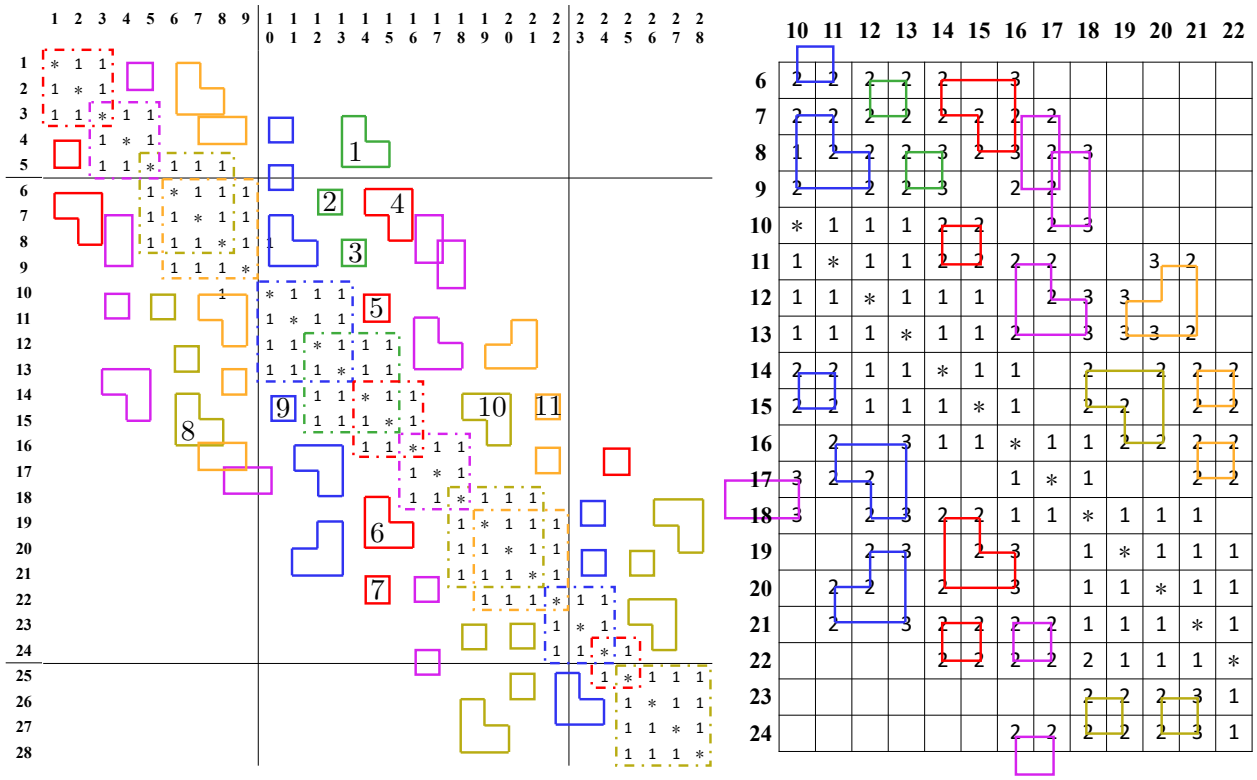


Figure 3.14: Cycles and the assignment for part of the cooperation matrix on the compatible graph of a symmetric cooperation.

matrix of $\{v_i\}_{10\leq i\leq 22}$.

Note that some sub-matrices of the cooperation matrix are marked in dashed colored rectangles; these sub-matrices are all square matrices and have all non-diagonal entries being ones. For any such rectangle, there are cycles marked in the same color (as the rectangle) that are totally contained within the columns spanned by this rectangle; these cycles are assigned a unique group number to form a group as specified in the previous subsection. In Fig. 3.15, instead of writing the group number assigned to each cycle, we mark the arrow connecting nodes representing the row and column indices of the cycle with a specified color for simplicity. Moreover, each one of those dashed rectangles corresponds to a maximum clique in $G(V, E)$ *that denotes the DSN.*

One can easily observe that cooperation levels assigned to entries in different columns within the same cycle are not always identical. Cooperation levels assigned to two nodes within the same column are identical if and only if they are on the same cycle or they are on different cycles from the same group.

Given all the aforementioned conditions, repeat steps 2)–5) specified in Example 9 to obtain a generator matrix of a cooperative coding scheme on the DSN in this example (Example 10). Then, for any node, the cross parities resulting from each cycle group can be derived from accessing other nodes in the maximum clique that contains this group. That is to say, by communicating with all the single-level neighbors, the cross parities for each cooperation level of any node $v_i \in V$ *are computable and can then be subtracted from the parity part of codeword* $\mathbf{c}_i$.

Take node v_{14} *as an example. The column representing* v_{14} *intersects with green cycles* C_1 *and* C_3*, and red cycles* C_4*,* C_5*,* C_6*, and* C_7*, corresponding to the* 3*-rd and the* 2*-nd level cooperation, respectively. All the nodes in the green clique* $\{v_{12}, v_{13}, v_{14}, v_{15}\}$ *except for* v_{14} *itself are locally-recoverable. Thus, their cross parities, resulting from the cooperation the green cycles represent, are computable, and they sum up to the* 3*-rd level cross parity of* $\mathbf{c}_{14}$. *Similarly, the* 2*-nd level cross parity of* $\mathbf{c}_{14}$ *can also be derived if the other two nodes in the*

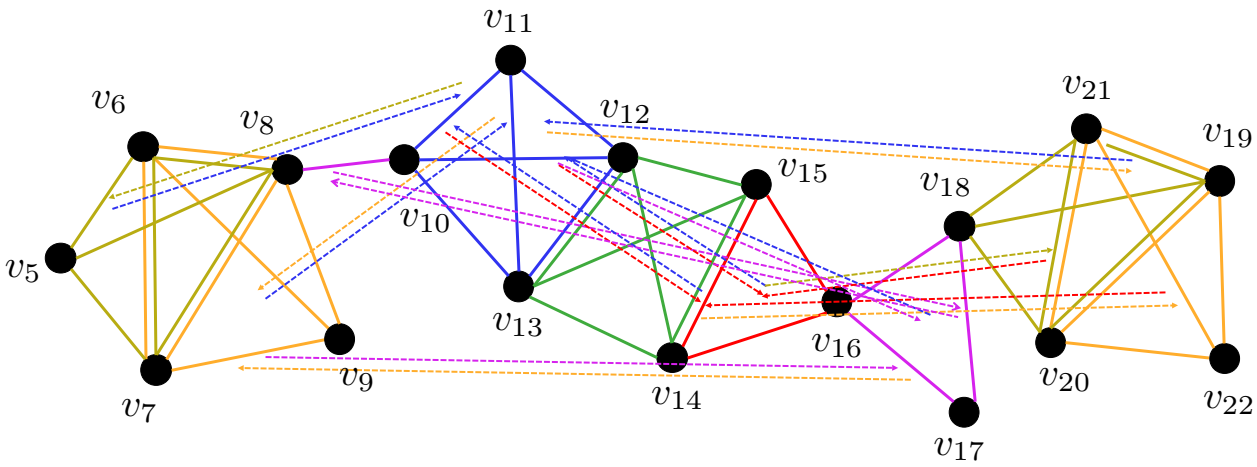


Figure 3.15: Cooperation graph of Example 10. For each node, all maximum cliques containing it are marked with different colors. For any dashed arrow pointing from an edge (or a triangle) to another edge (or another triangle), with the color identical to that of the maximum clique containing the latter one: it represents a cycle in the cooperation graph that is contained in the columns spanned by this maximum clique.

red clique $\{v_{14}, v_{15}, v_{16}\}$ *are locally-recoverable.*

More details, including the code construction, are given in Subsection 3.4.2 and Subsection 3.5.1.

Moreover, although Example 9 has a topologically symmetric cooperation graph and also a topologically symmetric compatible graph, it is not necessary in principle to constrain them to be symmetric. In the case where asymmetric cooperation is allowed, the basic components of the cooperation graph are edges instead of cycles, which allows more flexibility in choosing the cooperation graph. However, this asymmetry increases the complexity of defining the decoding graph for a node. Therefore, for simplicity, we only discuss topologically symmetric cooperations in this chapter.

3.4.2 Construction over Compatible Graphs

We have defined the notion of cooperation graphs in Section 3.4.1. Observe that the cooperation graphs shown in Fig. 3.12 and Fig. 3.15 satisfy a set of conditions that define the so-called **compatible graph**. We show in Theorem 4 the existence of a hierarchical coding

scheme with cooperation graph $\mathcal{G}$ if $\mathcal{G}$ is a compatible graph. The coding scheme is presented in Construction 5.

Definition 6. *For any graph $G(V, E)$ with $|V| = p$, a subgraph $G'(V', E')$ is called a **maximum clique** of G if any two nodes in V' are connected, and there does not exist any node in $V \setminus V'$ that is connected to all nodes in V'. The set of all maximum cliques of G is referred to as the **collection of maximum cliques** over G and is denoted by $\mathcal{S}(V, E)$. Each maximum clique in $\mathcal{S}$ is represented by a subset S of $[p]$, where S consists of the indices of all nodes in the maximum clique.*

Table 3.2 summarizes some notation associated with cooperation graphs that are used throughout the remainder of the chapter. Take the DSN and its cooperation matrix shown in Fig. 3.14 as an example. Observe that the green cycles in the columns spanned by the maximum clique $\{12, 13, 14, 15\}$ are indexed by 1, 2, and 3, and the red cycles in the columns spanned by the maximum clique $\{14, 15, 16\}$ are indexed by 4, 5, 6, and 7. These green and red cycles have group numbers 1 and 2, respectively. Suppose those cycles corresponding to the 2-nd level cooperation of v_{14} with top edges in the row representing v_{14} are labeled with 8, 9, 10, and 11. Note that these cycles and cycles C_4, C_5, C_6, and C_7 are symmetric with respect to the diagonal. Then, $T_1 = \{1, 2, 3\}$, $T_2 = \{4, 5, 6, 7\}$; $A_{14} = \{1, 2\}$; $S(1) = \{12, 13, 14, 15\}$, $S(2) = \{14, 15, 16\}$; $U_{14;1} = V_{14;3} = \{3, 4, 8, 9\}$; $U_{14;2} = V_{14;2} = \{6, 7, 10, 11, 18, 20, 21, 22\}$; $R_{14;2} = \{4, 5, 6, 7\}$; $T_{14;2} = \{8, 9, 10, 11\}$; $\ell_{5;14} = 2$; $\mathcal{M}_{14} = \mathcal{N}_{14} = \{v_{12}, v_{13}, v_{15}, v_{16}\}$. Definition 7 formally defines the sufficient conditions that result in a compatible graph, which were discussed informally in Example 10. Note that this definition of compatible graphs is more general than that presented in the short version of the chapter [34], since the cooperation levels of nodes on different columns of cycles are allowed to be different here.

Definition 7. *Let $\mathcal{G}$ be a cooperation graph on $G(V, E)$, where $\mathcal{G}$ is represented by $\{C_t = (X_t, Y_t, \{X_{t;j}\}_{j \in Y_t}, \{Y_{t;i}\}_{i \in X_t}, g_t, (\ell_{t;j})_{j \in Y_t})\}_{t \in [T]}$. Suppose node $v_i \in V$ has L_i cooperation levels. We call $\mathcal{G}$ a **compatible graph** on G if the following conditions are satisfied:*

Table 3.2: Notation associated with cooperation graphs.

Notation	Physical Interpretation
A	The total number of different cycle groups
T	The total number of different cycles
T_g	The set consisting of indices of cycles with group number g
A_i	The set of group numbers of those cycle groups that intersect with the column representing v_i
$S(g)$	The set consisting of vertices of the maximum clique that contains all columns spanned by cycles in group g
$U_{j;g}$	The intersection of all cycles contained in group g with the column representing v_j
$R_{j;\ell}$	The set consisting of indices of cycles that intersect with the column representing node v_j at its ℓ-th level cooperation
$V_{j;\ell}$	The intersection of all cycles contained in $R_{j;\ell}$ with the column representing v_j
$T_{i;\ell}$	The set consisting of indices of cycles that intersect with the row representing node v_i at its ℓ-th level cooperation (cycles with labels in $R_{j;\ell}$ and $T_{i;\ell}$ are symmetric with respective to the diagonal)
$\ell_{t;j}$	The cooperation level of node v_j in cycle C_t
$\gamma_{i;t}$	The maximum number of parity symbols that nodes in cycle C_t can provide to node v_i
$\mathcal{M}_i$	The set of nodes in the 1-st level cooperation graph of v_i
$\mathcal{N}_i$	The set of nodes in the neighborhood of v_i

1. *For any* $v_i \in V$, $\mathcal{M}_i \subseteq \mathcal{N}_i$.

2. *All cycles* C_t *with* $t \in [T]$ *are disjoint.*

3. *For any* $g \in [A]$, *there exists a maximum clique* $S(g) \in \mathcal{S}(V, E)$ *such that for all* $t \in T_g$, $Y_t \subseteq S(g)$.

4. *For each* $v_i \in V$, $\ell \in [L_i]$, *there exists a unique* $g \in A_i$, *such that* $V_{i;\ell} = U_{i;g}$; *denote* g *by* $g(i;\ell)$.

Construction 5. *Let* $G(V, E)$ *represent a DSN with parameters* $(\mathbf{n}, \mathbf{k}, \mathbf{r})$. *Suppose* $\mathcal{G}$ *is a compatible graph on* G, *with parameters* $\{C_t = (X_t, Y_t, \{X_{t;j}\}_{j\in Y_t}, \{Y_{t;i}\}_{i\in X_t}, g_t, (\ell_{t;j})_{j\in Y_t})\}_{t\in[T]}$. *Suppose node* $v_i \in V$ *has* L_i *cooperation levels.*

Let $\boldsymbol{\delta}$ *be the* 1*-st level cooperation parameter. For any* $v_i \in V$ *and* $g \in T_{i;\ell}$, *assign a cooperation parameter* $\gamma_{i;t} \in \mathbb{N}$ *to the cooperation between node* v_i *and nodes in* $Y_{t;i}$. *Let* $\eta_{j;\ell} = \max_{i\in V_{j;\ell}} \gamma_{i;t}$, *for* $\ell \in [L_i]$.

Let $u_i = k_i + \delta_i + \sum_{\ell=2}^{L_i} \eta_{i;\ell}$, $v_i = r_i + \sum_{v_j\in\mathcal{M}_i} \delta_j + \sum_{2\le\ell\le L_i, t\in T_{i;\ell}} \gamma_{i;t}$, *for* $i \in [p]$. *For each* $i \in [p]$, *let* $a_{i,s}$, $s \in [u_i]$, *and* $b_{i,t}$, $t \in [v_i]$, *be distinct elements of* $\mathrm{GF}(q)$, *where* $q \ge \max_{i\in[p]}\{u_i + v_i\}$.

Matrix $\mathbf{G}$ *in (3.1) is assembled as follows. Consider the Cauchy matrix* $\mathbf{T}_i$ *on* $\mathrm{GF}(q)^{u_i\times v_i}$ *such that* $\mathbf{T}_i = \mathbf{Y}(a_{i,1}, \ldots, a_{i,u_i}; b_{i,1}, \ldots, b_{i,v_i})$, *for* $i \in [p]$. *Then, we obtain* $\mathbf{A}_{i,i}$, $\mathbf{B}_{i,j}$, $\mathbf{E}_{i;\ell}$, $\mathbf{U}_i$, $\mathbf{V}_{i;\ell}$, *for* $i \in [p]$, $j \in [p] \setminus \{i\}$, $\ell \in [L_i]$, *according to the following partition of* $\mathbf{T}_i$:

$$\mathbf{T}_i = \left[\begin{array}{c|c|c|c|c} \mathbf{A}_{i,i} & \mathbf{B}_i & \mathbf{E}_{i;2} & \ldots & \mathbf{E}_{i;L_i} \\ \hline \begin{array}{c}\mathbf{U}_i\\ \hline \mathbf{V}_{i;2}\\ \hline \vdots\\ \hline \mathbf{V}_{i;L_i}\end{array} & \multicolumn{4}{c}{\multirow{4}{*}{$\mathbf{Z}_i$}} \end{array}\right], \tag{3.7}$$

$$\textit{where}\quad \mathbf{B}_i = \left[\begin{array}{c|c|c} \mathbf{B}_{i,j_1} & \ldots & \mathbf{B}_{i,j_{|\mathcal{M}_i|}} \end{array}\right], \tag{3.8}$$

$$\text{and } \mathbf{E}_{i;\ell} = \left[\begin{array}{c|c|c} \mathbf{E}_{i;\ell;t_1} & \dots & \mathbf{E}_{i;\ell;t_{|B_{i;\ell}|}} \end{array} \right], \tag{3.9}$$

such that $\mathcal{M}_i = \{v_{j_1}, v_{j_2}, \dots, v_{j_{|\mathcal{M}_i|}}\}$, $T_{i;\ell} = \{t_1, t_2, \dots, t_{|T_{i;\ell}|}\}$, $\mathbf{A}_{i,i} \in \mathrm{GF}(q)^{k_i \times r_i}$, $\mathbf{U}_i \in \mathrm{GF}(q)^{\delta_i \times r_i}$, $\mathbf{V}_{i;\ell} \in \mathrm{GF}(q)^{\eta_{i;\ell} \times r_i}$, $\mathbf{B}_{i,j} \in \mathrm{GF}(q)^{k_i \times \delta_j}$ *for all* $v_j \in \mathcal{M}_i^1$, *and* $\mathbf{E}_{i;\ell;t} \in \mathrm{GF}(q)^{k_i \times \gamma_{i;t}}$. *Let* $\mathbf{B}_{i,j} = \left[\mathbf{E}_{i;\ell;t}, \mathbf{0}_{k_i \times (\eta_{j;\ell} - \gamma_{i;t})}\right]$, *and* $\mathbf{A}_{i,j} = \mathbf{B}_{i,j}\mathbf{V}_{j;\ell}$, *for all* $j \in Y_{t;i}$, $t \in T_{i;\ell}$. *Let* $\mathbf{A}_{i,j} = \mathbf{B}_{i,j}\mathbf{U}_j$, *for* $v_j \in \mathcal{M}_i$; *otherwise* $\mathbf{A}_{i,j} = \mathbf{0}_{k_i \times r_i}$. *Substitute the components of* $\mathbf{G}$ *in (3.1).*

Let $\mathcal{C}_2$ *represent the code with generator matrix* $\mathbf{G}$.

Theorem 4. *The code* $\mathcal{C}_2$ *has EC hierarchies* $\mathbf{d}_i = (d_{i,0}, d_{i,1}, \dots, d_{i,L_i})$, *for all* $v_i \in V$, *where* $d_{i,0} = r_i - \delta_i - \sum_{\ell=2}^{L_i} \eta_{i;\ell}$, $d_{i,1} = r_i + \sum_{v_j \in \mathcal{M}_i} \delta_j$, *and* $d_{i,\ell} = r_i + \sum_{v_j \in \mathcal{M}_i} \delta_j + \sum_{2 \le \ell' \le \ell, t \in T_{i;\ell'}} \gamma_{i;t}$. *Moreover,* $\mathcal{I}_i^1 = \mathcal{M}_i$, $\mathcal{B}_i^1 = \bigcup_{v_j \in \mathcal{M}_i} (\mathcal{M}_j \setminus (\{v_i\} \cup \mathcal{M}_i))$. *For* $2 \le \ell \le L_i$, $\mathcal{I}_i^\ell = \bigcup_{t \in R_{i;\ell}} \{v_j : j \in X_{t;i}\} = \{v_j : j \in V_{i;\ell}\}$, $\mathcal{B}_i^\ell = \bigcup_{v_j \in \mathcal{I}_i^\ell} \left(\mathcal{I}_j^\ell \setminus (\{v_i\} \cup \mathcal{A}_i^\ell)\right)$ *(recall* $\mathcal{A}_i^\ell = \bigcup_{\ell' \le l} \mathcal{I}_i^{\ell'}$*),* $\lambda_{i,\ell;\mathcal{W}} = r_i + \sum_{j: v_j \in \mathcal{M}_i, (\mathcal{M}_j \setminus \{v_i\}) \subseteq (\mathcal{M}_i \cup \mathcal{W})} \delta_j + \sum_{\substack{(i,t): 2 \le \ell' \le \ell, t \in T_{i;\ell}, Y_{t;i} = \{j, j'\}, \\ \mathcal{I}_j^{\ell_{t;j}} \setminus \mathcal{A}_i^{\ell'} \subseteq (\{v_i\} \cup \mathcal{W}) \text{ or } \mathcal{I}_{j'}^{\ell_{t;j'}} \setminus \mathcal{A}_i^{\ell'} \subseteq (\{v_i\} \cup \mathcal{W})}} \gamma_{i;t}$, $\varnothing \subseteq \mathcal{W} \subseteq \mathcal{B}_i^\ell$.

Proof. For any node $v_j \in V$, denote the cross parities of v_j due to cooperation with nodes in $\mathcal{I}_j^\ell$ by $\mathbf{s}_{j;\ell}$, $\ell \in [L_j]$. The cross parities are given by the following equation:

$$\mathbf{s}_{j;\ell} = \begin{cases} \sum_{v_k \in \mathcal{M}_j} \mathbf{m}_k \mathbf{B}_{k,j}, & \ell = 1, \\ \sum_{k \in V_{j;\ell}} \mathbf{m}_k \mathbf{B}_{k,j}, & 2 \le \ell \le L_j. \end{cases} \tag{3.10}$$

Thus, the codeword stored at $v_j \in V$ can be expressed in the following form:

$$\begin{aligned} \mathbf{c}_j &= \mathbf{m}_j \mathbf{A}_{j,j} + \sum_{v_k \in \mathcal{M}_j} \mathbf{m}_k \mathbf{B}_{k,j} \mathbf{U}_j + \sum_{\ell=2}^{L_j} \sum_{k \in V_{j;\ell}} \mathbf{m}_k \mathbf{B}_{k,j} \mathbf{V}_{j;\ell} \\ &= \mathbf{m}_j \mathbf{A}_{j,j} + \mathbf{s}_{j;1} \mathbf{U}_j + \textstyle\sum_{\ell=2}^{L_j} \mathbf{s}_{j;\ell} \mathbf{V}_{j;\ell}. \end{aligned} \tag{3.11}$$

Provided that the rows of $\mathbf{A}_{j,j}$, $\mathbf{U}_j$, and $\{\mathbf{V}_{j;\ell}\}_{\ell=2}^{L_j}$ are linearly independent, $\mathbf{m}_j$ and $\{\mathbf{s}_{j;\ell}\}_{\ell \in [L_j]}$ are all computable if $\mathbf{c}_j$ is locally-recoverable.

We first show that by communicating with all the neighboring nodes in the 1-st level cooperation, the cross parities $\{\mathbf{s}_{i;\ell}\}_{\ell \in [L_i]}$ of any node $v_i \in V$ can be computed and removed

from the parity part of the codeword stored at this node if all its neighbors are locally-recoverable. Under this condition on the neighbors of v_i, calculating $\mathbf{s}_{i;1}$ is trivial. Next, we prove for $2 \leq \ell \leq L_i$ that $\{\mathbf{s}_{i;\ell}\}$ can also be computed.

Condition 4) in Definition 7 indicates that there exists a unique $g = g(i;\ell) \in A_i$, such that the following equation holds:

$$\mathbf{s}_{i;\ell} = \sum\nolimits_{j \in U_{i;g}} \mathbf{m}_j \mathbf{B}_{j,i}. \tag{3.12}$$

Moreover, Condition 3) guarantees the existence of a maximum clique $S(g) \in \mathcal{S}(V, E)$ such that for all $t \in T_g$, $Y_t \subseteq S(g)$. Let $\beta_{i;g} = \max_{\{(i,\ell):g(i;\ell)=g\}} \eta_{i;\ell}$, and $\mathbf{u}_{i;\ell} = \left[\mathbf{s}_{i;\ell}, \mathbf{0}_{\beta_{i;g}-\eta_{i;\ell}}\right]$, for all $i \in [p]$, $\ell \in [L_i]$. We now consider:

$$\begin{aligned}
\sum_{(i,\ell):g(i;\ell)=g} \mathbf{u}_{i;\ell} &= \sum_{t \in T_g, i \in X_t, j \in Y_{t;i}} \left[\mathbf{m}_i \mathbf{B}_{i,j}, \mathbf{0}_{\beta_{i;g}-\eta_{i;\ell_{t;i}}}\right] \\
&= \sum_{t \in T_g, i \in X_t, j \in Y_{t;i}} \left[\mathbf{m}_i \mathbf{E}_{i;\ell_{t;i};t}, \mathbf{0}_{\beta_{i;g}-\gamma_{i;t}}\right] \\
&= \sum_{t \in T_g, i \in X_t} \mathbf{0}_{\beta_{i;g}} = \mathbf{0}_{\beta_{i;g}}.
\end{aligned}$$

It follows that $\mathbf{s}_{i;\ell}$ can be derived from $\mathbf{u}_{i;\ell}$ if all $\mathbf{s}_{j;\ell'}$ such that $g(j;\ell') = g$ are known. Condition 3) in Definition 7 implies that all these j's belong to $S(g)$, and the set of nodes indexed by $S(g)$ is a subset of $\mathcal{M}_i$, which means that all the aforementioned $\mathbf{s}_{j;\ell'}$'s are computable given that the neighbors of v_i are locally-recoverable.

We have proved that all the ℓ-th level cross parities, $\ell \in [L_i]$, of any node $v_i \in V$ can be computed if the neighboring nodes are locally-recoverable. Now, we move forward to calculate the EC hierarchies of each node. Observe that the local and the 1-st level cooperation erasure correction capabilities are proved the same way they are proved for Theorem 2. Thus, we only consider cases where $2 \leq \ell \leq L_i$ in the following graph.

Moreover, for $2 \leq \ell \leq L_i$, any cycle C_t with index $t \in T_{i;\ell}$ has a potential to provide additional $\gamma_{i;t}$ in the ℓ-th level cross parities to v_i. Therefore, $d_{i,\ell} = r_i + \sum_{v_j \in \mathcal{M}_i} \delta_j +$

$\sum_{2\leq\ell'\leq\ell,t\in T_{i,\ell'}}\gamma_{i;t}$. The term $\gamma_{i;t}$ is added to the EC capability if any one of the two nodes v_j and $v_{j'}$, $Y_{t;i}=\{j,j'\}$, obtains its $\gamma_{i;t}$ cross parities at v_i through its $\ell_{t;j}$-th or $\ell_{t;j'}$-th level cooperation. Namely, any one of $\mathbf{m}_i\mathbf{B}_{i,j}$ and $\mathbf{m}_i\mathbf{B}_{i,j'}$ provides the value of $\mathbf{m}_i\mathbf{E}_{i;\ell_{t;i},t}$, and thus provides extra $\gamma_{i;t}$ parity symbols to v_i. Provided that $\mathbf{s}_{j;\ell_{t;j}}$ can be computed if v_j is locally-recoverable, one needs to know the cross parities from all the nodes in the set $\mathcal{I}_j^{\ell_{t;j}}\setminus\{v_i\}$ to obtain those extra $\gamma_{i;t}$ parity symbols, i.e., those nodes are locally-recoverable, which means $\mathcal{I}_j^{\ell_{t;j}}\setminus\mathcal{A}_i^{\ell}\subseteq\{v_i\}\cup\mathcal{W}$. Similarly, the condition of $v_{j'}$ successfully calculating these $\gamma_{i;t}$ cross parities at v_i is described as $\mathcal{I}_{j'}^{\ell_{t;j'}}\setminus\mathcal{A}_i^{\ell}\subseteq\{v_i\}\cup\mathcal{W}$. Therefore, the overall requirement is stated as "$\mathcal{I}_j^{\ell_{t;j}}\setminus\mathcal{A}_i^{\ell}\subseteq\{v_i\}\cup\mathcal{W}$ or $\mathcal{I}_{j'}^{\ell_{t;j'}}\setminus\mathcal{A}_i^{\ell}\subseteq\{v_i\}\cup\mathcal{W}$". From this discussion, we reach that $\mathcal{B}_i^{\ell}=\bigcup_{t\in T_{i,\ell},j\in Y_{t;i}}(\mathcal{I}_j^{\ell_{t;j}}\setminus(\{v_i\}\cup\mathcal{A}_i^{\ell}))$ and $\lambda_{i,\ell;\mathcal{W}}=r_i+\sum_{j:v_j\in\mathcal{M}_i,(\mathcal{M}_j\setminus\{v_i\})\subseteq(\mathcal{M}_i\cup\mathcal{W})}\delta_j+\sum_{\substack{(i,t):2\leq\ell'\leq\ell,t\in T_{i;\ell'},Y_{t;i}=\{j,j'\},\\ \mathcal{I}_j^{\ell_{t;j}}\setminus\mathcal{A}_i^{\ell'}\subseteq(\{v_i\}\cup\mathcal{W})\text{ or }\mathcal{I}_{j'}^{\ell_{t;j'}}\setminus\mathcal{A}_i^{\ell'}\subseteq(\{v_i\}\cup\mathcal{W})}}\gamma_{i;t}$, for $\varnothing\subseteq\mathcal{W}\subseteq\mathcal{B}_i^{\ell}$. □

Note that although in a DSN represented by $G(V,E)$, node $v_i\in V$ cooperates with all nodes in $\mathcal{I}_i^{\ell}$ in the ℓ-th level cooperation, $2\leq\ell\leq L_i$, it is not necessary that all codewords stored in nodes from $\mathcal{I}_i^{\ell}$ need to be recovered. The reason is that these nodes are partitioned into node pairs where the two nodes in the same pair provide exactly the same group of parity symbols and only one of them needs to be recovered for node v_i to recover its codeword, as we discussed in Theorem 4.

For example, suppose Fig. 3.13 corresponds to a subgraph of a DSN with a recoverable erasure pattern. The pink triangles and the dashed arrows represent the ℓ-th level cooperation at each node such that no other nodes are involved in the ℓ-th level cooperation of these nodes, i.e., C_t is the only cycle in the cycle group containing it. As discussed in Theorem 4, to remove the local cross parities of each node, neighbors of any non-locally-recoverable node should all be locally-recoverable. Therefore, there exists at most one non-locally-recoverable node in each one of the two triangles. For any $i\in X_t$, previous conditions indicate that at least one of the two nodes with indices in $Y_{t;i}$ is locally-recoverable; let it be node v_j, where $j\in Y_{t;i}$. We know that $Y_{t;j}$ consists of i and i' for some $i'\in X_t$ and $i'\neq i$. Since the code-

word stored at v_j is locally-recoverable and C_t forms an isolated cycle group, the cross parity $\mathbf{m}_i\mathbf{B}_{i,j} + \mathbf{m}_{i'}\mathbf{B}_{i',j}$ can be derived at node v_j. Since the codeword at $v_{i'}$ is locally-recoverable, $\mathbf{m}_{i'}\mathbf{B}_{i',j}$ can be further subtracted from the cross parity to obtain $\mathbf{m}_i\mathbf{B}_{i,j}$. This observation indicates that regardless of the way the matching graph is specified and the indices of the nodes that are not recovered, the non-recovered nodes are able to obtain their ℓ-th level cross parities. For example, suppose then v_{i_1} and v_{j_1} in Fig. 3.13 are not locally-recoverable. Since $V_{i_1;\ell} = \{j_1, j_3\}$ and $\mathcal{I}_i^\ell = \{v_{j_1}, v_{j_3}\}$, the aforementioned discussion demonstrates that recovering v_{i_1} does not require v_{j_1} to be recovered.

In Fig. 3.13, the additional erasure correction capabilities offered to the nodes are identical regardless of the structure of the local matching graph. However, this may not be true in general, if more than one cycle gets involved. In general, different local matching graphs are likely to result in non-identical erasure correction capabilities. In particular, although the EC hierarchies are defined by $(\lambda_{i,l;\mathcal{W}})_{\varnothing\subset\mathcal{W}\subset\mathcal{B}_i^\ell}$ for each individual node at each cooperation level, this can be more elaborately defined since accessing a different subset of nodes in $\mathcal{I}_i^l$ may result in different EC capabilities even if $\mathcal{W}$ are the same, and $\lambda_{i,l;\mathcal{W}}$ only specifies the largest one. We show it in details by Example 11.

Given an isolated matching graph G', if none of the non-locally-recoverable nodes are able to derive any additional cross parities solely from the local cooperation specified by G', i.e., all cycles in G' constitute a cycle group, we call it an **absorbing matching graph**.

In Example 11, EC capabilities of compatible graphs resulting from the same cooperation graph associated with different local matching graphs, as shown in Fig. 3.16, are discussed. We prove that the left two panels are two absorbing matching graphs, while the right two panels are not, which also demonstrates that the EC capability is not uniquely determined by the cooperation graph. Instead, the local matching graph also matters.

We focus on the local cooperation graph between 9 nodes in a DSN $G(V, E)$, where $V = \{v_i\}_{i\in[9]}$. Let $V_1 = \{v_1, v_4, v_7\}$, $V_2 = \{v_2, v_5, v_8\}$, and $V_3 = \{v_3, v_6, v_9\}$, and suppose nodes in each one of these sets mutually cooperate with nodes in each of the remaining two

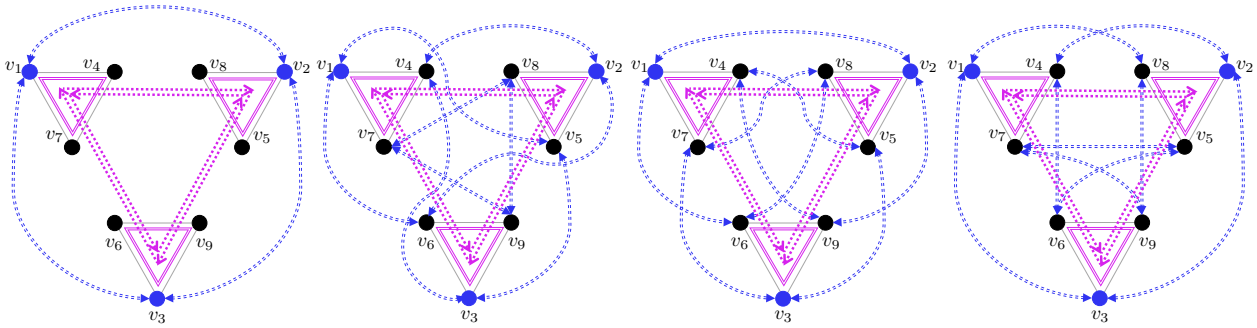


Figure 3.16: Possible local matching graphs contained in a multi-level cooperation graph between 9 nodes. The nodes are partitioned into three groups, where nodes within each one of them are pairwisely connected. Each dashed double-sided arrow represent a cycle.

sets. According to the definition of cooperation graphs, each one of the graphs represents 3 cycles, $\{C_i\}_{i\in[3]}$, where $X_i = \{3j+i\}_{0\leq j\leq 2}$, $Y_i = X_{i+1}$, and $X_4 = X_1$. Suppose $\{C_i\}_{i\in[3]}$ form an isolated cycle group in the cooperation graph on G. Represent each one of the cycles by a specified matching graph and refer to the resulting local matching graph as an isolated local matching graph. Fig. 3.16 presents four different isolated local matching graphs on these nodes. Let colors blue and black refer to nodes that are non-locally-recoverable and locally-recoverable nodes, respectively. Then, each V_i, $i \in [3]$, contains at most one blue node if it is contained in a recoverable erasure pattern.

Example 11. (Absorbing Matching Graphs) *Consider the left-most local matching graph in Fig. 3.16, we notice that v_1, v_2, v_3 are mutually connected (and we will show that the connections between the rest of the nodes actually do not matter, thus we omit them in the figure). Without loss of generality, it is sufficient to prove that v_1 is not able to obtain any extra parity symbols from cycle C_1. Since v_1 and v_2 are connected, we know that $Y_{1;1} = \{5, 8\}$. Thus, v_1 needs to either obtain $\mathbf{m}_1\mathbf{B}_{1,5}$ from v_5 or $\mathbf{m}_1\mathbf{B}_{1,8}$ from v_8. Since v_2 and v_3 are connected, both v_5 and v_8 have cross parities at node v_3. Therefore, the parities v_i, $i \in \{5, 8\}$, are of the form $\mathbf{s}_i = \mathbf{m}_1\mathbf{B}_{1,i} + \mathbf{m}_3\mathbf{B}_{3,i} + \mathbf{m}_j\mathbf{B}_{j,i} + \mathbf{m}_{j'}\mathbf{B}_{j',i}$, where $j \in \{4, 7\}$, $j' \in \{6, 9\}$. If codewords stored at node v_j and $v_{j'}$ are all locally-recoverable, their parities can be subtracted from $\mathbf{s}_i$ to obtain the remainder $\mathbf{s}'_i = \mathbf{m}_1\mathbf{B}_{1,i} + \mathbf{m}_3\mathbf{B}_{3,i}$. Observe that in*

order to obtain $\mathbf{m}_1\mathbf{B}_{1,i}$, $\mathbf{m}_3\mathbf{B}_{3,i}$ *needs to be obtained first. That is to say,* v_1 *can only obtain additional parities unless the message in* v_3 *is recovered. However, following a similar process, we will need* v_2 *to be recovered for* v_3*, and* v_1 *to be recovered for* v_2*. This cyclic requirement indicates that* v_1*,* v_2*, and* v_3 *are "absorbed" into a balanced situation where none of them can be recovered first, which cannot be broken unless information from the rest of the graphs is provided.*

Similarly, we can prove that the second-to-the-left panel is also an absorbing matching graph. Since the connections between the three triangles are symmetric, it is still sufficient to prove that v_1 *is not able to obtain any extra parity symbols from cycle* C_1*. Since* v_1 *and* v_5 *are connected, we know that* $Y_{1;1} = \{2, 8\}$*. Given that* v_2 *is not locally-recoverable, the only path for* v_1 *to obtain extra parities is to obtain* $\mathbf{m}_1\mathbf{B}_{1,8}$ *from* v_8*. Observe that* v_8 *is connected to* v_7 *and* v_9 *in the matching graph, which means that* v_8 *has cross parities at* v_1*,* v_3*,* v_4*, and* v_6*. Therefore,* v_8 *needs* v_3*,* v_4*, and* v_6 *to be all recovered in order to subtract* $\mathbf{m}_3\mathbf{B}_{3,8}$*,* $\mathbf{m}_4\mathbf{B}_{4,8}$ *and* $\mathbf{m}_6\mathbf{B}_{6,8}$ *from* $\mathbf{m}_3\mathbf{B}_{3,8} + \mathbf{m}_4\mathbf{B}_{4,8} + \mathbf{m}_6\mathbf{B}_{6,8}$ *to obtain* $\mathbf{m}_1\mathbf{B}_{1,8}$*. This requires* v_3 *to be recovered. Following a similar argument to that of the left-most panel, this graph is also an absorbing matching graph.*

Moving on to the second-from-the-right panel, v_5 *has cross parities at* v_7*,* v_6*,* v_9*, and* v_1*. Therefore,* v_1 *is able to obtain the cross parities resulting from cooperation cycle* C_3 *through* v_5*. Similarly,* v_2 *is able to obtain the cross parities resulting from cooperation cycle* C_2 *through* v_4*. Finally,* v_3 *is able to obtain its cross parities from* C_3 *and* C_2*.*

Now we look at the right-most panel, v_4 *has cross parities at* v_5*,* v_8*,* v_9*, and* v_3*. Therefore,* v_3 *is able to obtain the cross parities resulting from cooperation cycle* C_3 *through* v_4*. Similarly,* v_3 *is also able to obtain the cross parities resulting from cooperation cycle* C_2 *through* v_5*. After that, one of* v_1 *and* v_2 *is able to obtain additional parities from* C_1*, and the other one can obtain additional parities from both* C_1 *and its cooperation with* v_6 *and* v_9*.*

Note that the major difference between the second-to-the-right and the right-most panel is that the decoding of v_1 *provides no additional parities on* v_2 *and* v_3 *in the third one.*

Therefore, if each cycle C_i, $i \in [3]$, provides η cross parities at each of its nodes, the third and the fourth graphs allow up to additional 4η and 5η cross parities, respectively.

We have shown in the previous example that the left-most two panels in Figure 3.13 are absorbing matching graphs, and they become non-absorbing matching graphs if any of the blue nodes turns to be recovered from cooperation with the rest of the graph. In this case, without loss of generality, suppose v_3 is recovered, then v_1 in these two graphs is also recoverable according to discussion in Example 11. While in the left-most panel, nodes in $\mathcal{I}_1^l$ are indeed all locally recovered, those in the second graph from the left are not. We also notice that in the right-most panel, nodes in $\mathcal{I}_1^l$ are locally recovered to recover v_1, while those in the second-to-the right panel are not. Given that any isolated matching graph corresponding to a $\lambda_{1,l;\mathcal{W}}$ with $\mathcal{W} = \varnothing$, this example also demonstrates that a different set $\mathcal{A}_i^l$ will also provide different EC capabilities. While in Example 11 we already subtly discussed such a scenario, we leave more detailed analysis for future work.

Moreover, we state without proof here that the left-most two panels of Fig. 3.16 discussed in Example 11 are all the possible structures of an absorbing matching graph for this specified local cooperation graph (subject to the graph isomorphism). Since these graphs also are all the possible structures where the nine matching edges form disconnected cycles, those edges in other matching graphs all form a cycle of length 9 and are mutually isomorphic according to permutations of v_1 to v_9. However, different permutations of the nodes do result in different erasure correction capabilities. For example, the second-to-the-right panel is isomorphic to the right-most panel if v_3, v_5, v_7 are blue instead, as shown in Figure 3.17. This has no impact on the average erasure correction capability while looking into the local matching graphs individually, but the permutation matters while taking the connection to the rest of the graphs into consideration.

In Remark 2, we discuss the information flow between neighboring nodes, and the information flow between nodes with distance two through their common neighbors. Observe that nodes cooperating with any given node in its higher-level cooperations are not neces-

sarily all within its two-hop neighborhood. However, these nodes actually provide additional parities to the original nodes. This scenario is not covered by the previous definition of information flow, instead of it, we proposed the notion of **information coupling** to describe it, as discussed in Remark 3.

Remark 3. (Information Coupling in Multi-Level Coded DSN) *Take the local matching graph shown in the right-most panel in Fig. 3.13 as an example. Consider the case where non-locally-recoverable nodes are v_1, v_2 and v_3, as shown in the left panel in Fig. 3.17. Node v_3 is able to obtain additional parities from v_4 since v_5, v_8 and v_9 are locally-recoverable. This case can be regarded as information flow from the cooperation between v_4, v_5 and v_8 to the cooperation between v_4, v_3 and v_9 through v_4.*

Consider another case where the non-locally-recoverable nodes are v_3, v_5 and v_7 instead, as shown in the right panel in Fig. 3.17. Node v_3 is no longer able to decode its codeword first. Instead, node v_7 is able to obtain additional parities from v_2 since v_1, v_6 and v_9 are locally-recoverable. This case can be regarded as information flow from the cooperation between v_1, v_2, and v_7 to the cooperation between v_2, v_6, and v_9 through v_2.

The aforementioned cases indicate that for any node, nodes cooperating with it in its higher-level cooperation do not have impact on it individually, but rather collectively. Moreover, as discussed in Example 11, this impact is not only dependent on the local matching graphs, but also dependent on the erasure patterns. Therefore, instead of discussing information flow between two cycles, it is more appropriate to treat all the cycles contained in any cycle group collaboratively. This can be interpreted as information coupling resulted from the cooperation between $\{v_1, v_4, v_7\}$, $\{v_2, v_5, v_8\}$, and $\{v_3, v_6, v_9\}$, as an analogy to information coupling in network navigation.

3.4.3 Recoverable Erasure Patterns

Recall the notion of "decoding graph" in Definition 4, under which recoverable erasure patterns of the single-level cooperative codes are described. However, in cases where higher-level

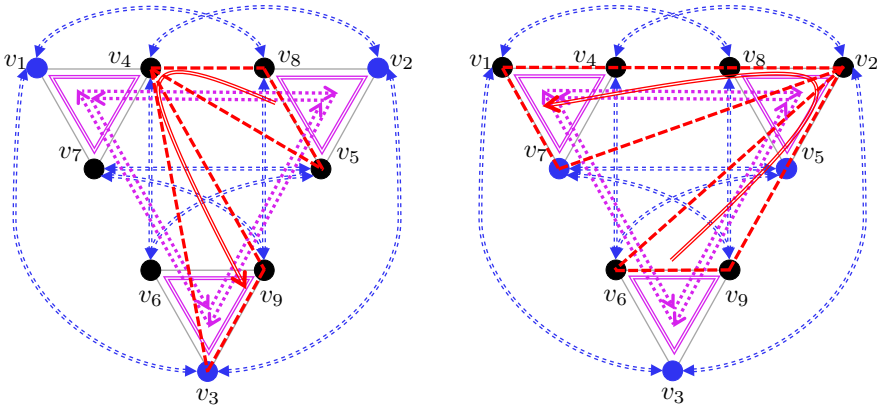


Figure 3.17: Information coupling. The two graphs represent two different erasure patterns for the non-absorbing local matching graphs in Fig. 3.13. While the information flow depicts the communication between any two nodes separated by a distance of 1 or 2 in their 1-st level cooperation, it is not able to fully describe higher level cooperations. Two cycles in a local matching graph also help the decoding of nodes on each other through their shared nodes, and we call this "information coupling".

cooperations are involved, Definition 4 is not enough to define and enumerate all associated recoverable erasure patterns. In this section, we extend Definition 4 into Definition 8 to allow for the multi-level cooperation. Recoverable erasure patterns of hierarchical codes are specified in Theorem 5.

Definition 8. (Decoding Graph in Multi-Level Cooperation) *Let $G(V, E)$ represent a DSN with $|V| = p$. Let $\mathcal{T}(\mathcal{V}, \mathcal{E})$ denote a directed subgraph of G. For all $v_i \in \mathcal{V}$, denote the set containing the children of v_i by $\mathcal{V}_i^{\mathrm{C}}$, and the set containing all parents of v_i by $\mathcal{V}_i^{\mathrm{P}}$. Suppose v_j is the only node without parents, we call it the* ***root*** *of $\mathcal{T}$. We call any node without children a* ***leaf****. Suppose that all the leaves of $\mathcal{T}$ are not locally-recoverable, and any other $v_i \in \mathcal{V}$ satisfies either one of the following conditions.*

1. *The codeword stored at v_i is locally-recoverable: there exists a set $\mathcal{L} \subseteq \{2, \ldots, L_i\}$, with $|\mathcal{V}_i^{\mathrm{P}} \cap \mathcal{M}_i| \in \{0, 1\}$, $|\mathcal{V}_i^{\mathrm{P}} \cap V_{i;\ell}| = 1$, and $\mathcal{V}_i^{\mathrm{P}} \cup \mathcal{V}_i^{\mathrm{C}}$, that consists of all nodes with indices in $V_{i;\ell}$ for $\ell \in \mathcal{L}$ (and $\mathcal{M}_i$ if $|\mathcal{V}_i^{\mathrm{P}} \cap \mathcal{M}_i| = 1$), where codewords stored at them are not locally-recoverable.*

2. *The codeword stored at v_i is not locally-recoverable: codewords stored at nodes from*

$\mathcal{V}_i^{\mathrm{P}} \cup \mathcal{V}_i^{\mathrm{C}}$ *are locally-recoverable.*

We call $\mathcal{T}$ *a decoding graph at its root node* v_j *over* $G(V, E)$*.*

Theorem 5. (Flexible Erasure Patterns) *Let* $\mathcal{C}$ *be a code with hierarchical cooperation on a DSN represented by* $G(V, E)$*, where* $\mathcal{C}$ *and all related parameters are specified according to Construction 5. Let* $\mathbf{u} \in \mathbb{N}^p$ *such that* $\mathbf{u} \preceq \mathbf{n}$*. Suppose* $\mathcal{C}$ *and* $\mathbf{u}$ *satisfy the following conditions:*

1. *Let* V^{NL} *represent the set contains all the nodes* v_i*,* $i \in [p]$ *such that* $u_i > r_i - \delta_i$*. Let* $V^{\mathrm{L}} = V \setminus V^{\mathrm{NL}}$*. Then, for any* $v_i \in V^{\mathrm{NL}}$*,* $\mathcal{M}_i \subset V^{\mathrm{L}}$*.*

2. *For any* $v_i \in V^{\mathrm{NL}}$*, there exists a decoding graph* $\mathcal{T}_i(\mathcal{V}_i, \mathcal{E}_i)$ *at root* v_i *over* G*. Moreover, for any leaf* v_j *of* $\mathcal{T}_i$*,* $u_j \leq r_j$*; for any node* $v_j \in \mathcal{V}_i \cap V^{\mathrm{NL}}$*,* $u_j \leq r_j + \sum_{v_k \in \mathcal{M}_i \cap \mathcal{V}_j^{\mathrm{C}}} \delta_k + \sum_{\ell=2}^{L_i} \sum_{k \in V_{j;\ell} \cap X_{t;j}, v_k \in \mathcal{V}_j^{\mathrm{C}}} \gamma_{k;t}$*.*

Then, $\mathbf{u}$ *is a* **recoverable erasure pattern** *of* $\mathcal{C}$ *over* $G(V, E)$*.*

Consider the DSN with the 1-st level cooperation graph presented in Example 7. We add the 2-nd level and the 3-rd level cooperation graphs to the DSN and mark them in pink and olive, respectively, as shown in Example 12 and Figure 3.19. Black and blue/green still refer to nodes where the stored codewords are locally-recoverable and non-locally-recoverable, respectively. Components marked in red represent local decoding graphs, which is the subgraph of the decoding graph corresponding to the local matching graph.

Example 12. *Fig. 3.18 has five graphs. The left-most panel describes the cooperation graph resulting from adding the 2-nd level cooperation among nodes to the DSN in Example 7, where there exist two possible local matching graphs that are specified by the two graphs in the center (in the central panel). The right-most two panels present the subgraphs in local decoding graphs corresponding to the two possible local matching graphs. Let* $\mathbf{u} = (u_1, u_2, \ldots, u_p)$ *be an erasure pattern on this DSN. Under the EC solution specified in Theorem 4, suppose there exists* $\gamma \in \mathbb{N}$ *such that* $\gamma_{i;t} = \gamma$*, for all* $t \in [T]$*, and* $i \in X_t$*.*

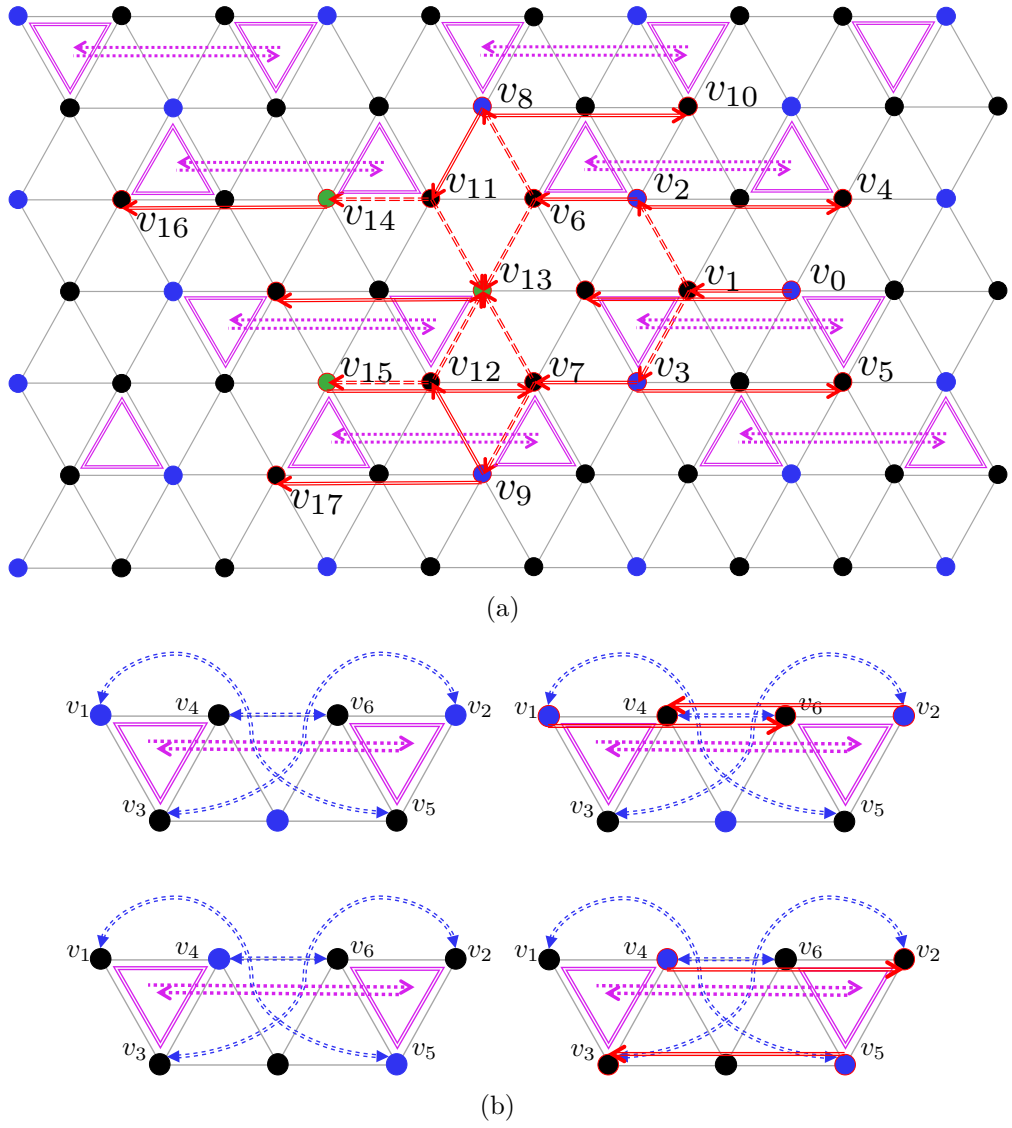
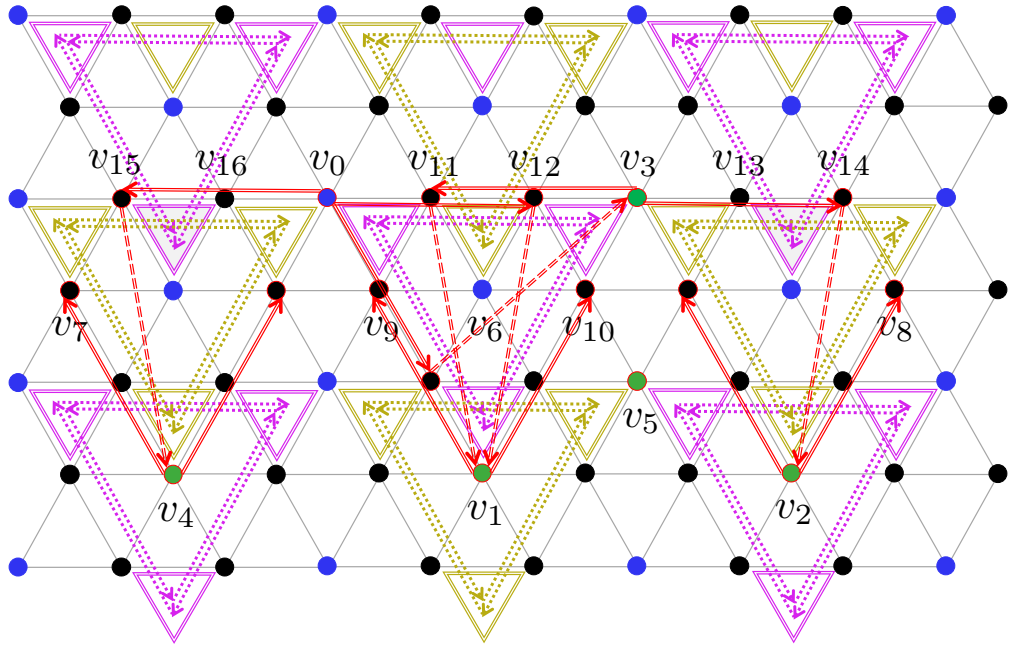


Figure 3.18: DSN (a) and the local matching graph (b) specified for 2-nd level cooperation in Example 12.

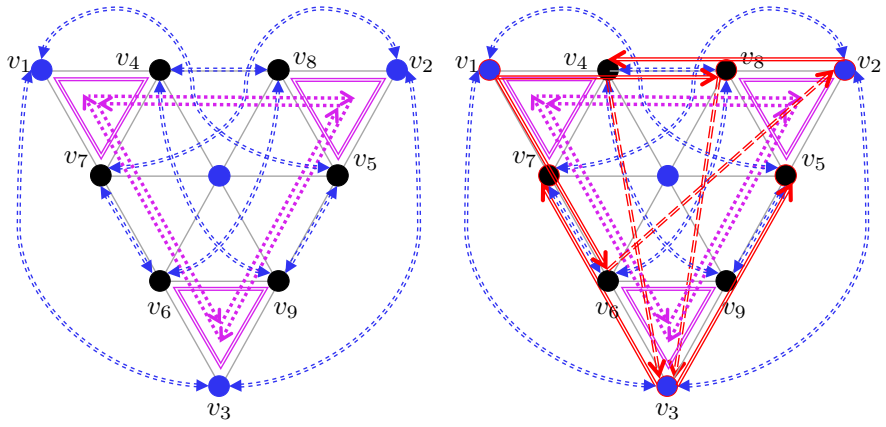
In the specified cooperation graph, for $i \in [p]$, if v_i is black, then $0 \leq u_i \leq r_i - \delta - \gamma$; else if v_i is green, then $r_i - \delta - \gamma \leq u_i \leq r_i + \gamma$; otherwise $r_i + \gamma < u_{1,i} \leq r_i + \delta + \gamma$. Since each blue node is contained in an isolated local matching graph, it can obtain additional γ cross parity symbols from its 2-nd level cooperation according to the previous discussion about Fig. 3.13. Therefore, all the non-locally-recoverable nodes are able to tolerate extra γ erasures, which means that $\mathbf{u}$ is a recoverable erasure pattern of this graph but not a recoverable pattern of the left panel in Fig. 3.8.

Example 13. *Fig. 3.19 has three graphs. The left-most one describes the cooperation graph resulting from adding the 2-nd and the 3-rd level cooperations among nodes to the DSN in Example 7. We adopt the right-most local matching graph in Fig. 3.16 to specify local matching graphs in this example, and it is shown in the central panel. Note that we have exchanged the indices of v_4 and v_7, and those of v_5 and v_8 in the original graph to obtain the graph in the center. The right-most panel presents the subgraph in local decoding graphs corresponding to the local matching graph. Let $\mathbf{u} = (u_1, u_2, \ldots, u_p)$ be an erasure pattern on this DSN. Under the EC solution specified in Theorem 4, suppose there exists $\gamma \in \mathbb{N}$ such that $\gamma_{i;t} = \gamma$, for all $t \in [T]$, and $i \in X_t$.*

In the specified cooperation graph, for $i \in [p]$, if v_i is black and is connected to two triangles, then $0 \leq u_i \leq r_i - \delta - 2\gamma$; else if v_i is black and is connected to only one triangle, then $0 \leq u_i \leq r_i - \delta - \gamma$; else if v_i is blue and is connected to only one triangle, $r_i - \delta - \gamma < u_{1,i} \leq r_i + 2\gamma$; else if v_i is green, then $r_i - \delta - 2\gamma < u_i \leq r_i + 2\gamma$; otherwise $r_i - \delta - 2\gamma < u_{1,i} \leq r_i + 3\gamma$. Since each non-locally-recoverable node, e.g., v_1, v_2, and v_4, is contained in an isolated local matching graph, i.e., a triangle, laying at the bottom of this triangle, it can obtain additional 2γ cross parity symbols from it according to the previous discussion in Example 11. Then, v_3 and v_5 can also obtain extra 2γ parity symbols, where γ of them are from the 2-nd level cooperation, and the remaining γ of them are from the 3-rd level cooperation, respectively, according to Example 11. After that v_0 is able to obtain 2γ cross parity symbols from the 2-nd level cooperation (the pink triangle), and γ cross parity



(a)



(b)

Figure 3.19: DSN (a) and the local matching graph (b) specified for 2-nd and 3-rd level cooperation in Example 13.

symbols from the 3*-rd level cooperation (the olive triangle). Following a similar logic, all codewords in the non-locally-recoverable nodes are able to be recovered, which means that* $\mathbf{u}$ *is a recoverable erasure pattern of this graph but not a recoverable pattern of any of the graphs in Fig. 3.8.*

3.5 Topology Adaptivity, Scalability, and Flexibility

In Section 3.4, we have presented a construction of codes with hierarchical locality for a DSN with a given cooperation graph, which enables the system to offer multi-level access at each node while simultaneously reducing the latency by taking into account the communication cost between different nodes. However, multi-level accessibility is not the only property that is desirable in practical cloud storage applications. In this section, we therefore discuss topology adaptivity, scalability, and flexibility of our construction, which are especially critical in dynamic cloud storage.

3.5.1 Topology Adaptivity

As discussed in Section 4.1, varying topology is a critical property of DSNs because of the dynamic nature of practical networks. While discussing EC solutions for DSNs with a specific topology, the time cost in each communication link and the erasure statistics of each node should also be taken into consideration to have a good trade-off between low latency and high EC capability. Although hierarchical coding schemes over a DSN with a specified cooperation graph has been discussed in Subsection 3.4.1, the method of finding a cooperation graph over DSNs with arbitrary topology has not yet been discussed. Algorithm 1 searches for a cooperation graph over a given network; the existence of such a graph is implicitly proved in the algorithm. Here $G(V, E)$ denotes a DSN with the collection $S(V, E)$ of maximum cliques.

The complexity of Algorithm 1 is roughly calculated as follows. Suppose the maximum degree of each node is D, and the maximum size of a clique is bounded by some constant

(which is true in sparse graphs), then the complexity of finding all the cliques is $O(|V|D)$. Moreover, the algorithm needs additional $O\left(\sum_{g\in a(S)} b(g)\right)$ operations to set the indices of all the cycles. Therefore, the overall complexity is $O\left(|V|D + \sum_{g\in a(S)} b(g)\right)$, which is approximately linear in the network size when the network graph is sparse.

Remark 4. *(Latency Optimization in Cooperation Graphs) One might observe that although Algorithm 1 presents a general method to search for a cooperation graph over a given DSN described by $G(V, E)$, the resulting code is not guaranteed to possess optimized latency. Optimization of the construction with the lowest latency is left for future work.*

3.5.2 Scalability

As discussed in Section 4.1, scalability refers to the capability of expanding the backbone network to accommodate additional workload without rebuilding the entire infrastructure. More specifically, when a new cloud is added to the existing configuration, computing a completely different generator matrix results in changing all the encoding-decoding components in the system, and is very costly. The preferred scenario is that adding a new cloud does not change the encoding-decoding components of the existing clouds.

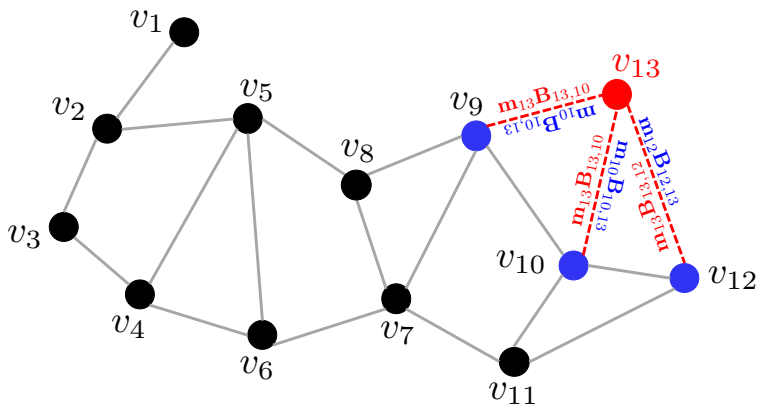


Figure 3.20: Scalability: Add a node to existing DSN.

We show that our construction naturally achieves this goal. For simplicity, we only discuss the scalability over constructions with single-level cooperation here. Observe that in

Algorithm 1 Cooperation Graph Search

Inputs:

$G(V, E)$: existing DSN;

$a(S)$: the number of different cycle groups associated with the maximum clique $S \in S(V, E)$;

$b(g)$: the number of cycles within the cycle group g;

Outputs:

$\mathcal{G}(\mathcal{V}, \mathcal{E})$: a cooperation graph over G;

//Find a cooperation graph

$\mathcal{V} \leftarrow V$, $\mathcal{E} \leftarrow \varnothing$;
for $v_i \in V$ **do**
    Assign a subset of $\mathcal{N}_i$ to $\mathcal{M}_i$;
    $\mathcal{E} \leftarrow \mathcal{E} \cup \{e_{i,j} : v_j \in \mathcal{M}_i\}$;
end for
Find the collection $\mathcal{S}(\mathcal{V}, \mathcal{E})$ of maximum cliques over $\mathcal{G}$;
$t \leftarrow 1$, $g \leftarrow 1$;
for $S \in \mathcal{S}(\mathcal{V}, \mathcal{E})$ **do**
    for $1 \leq i \leq a(S)$ **do**
        for $1 \leq b \leq b(g)$ **do**
            Find an edge or a triangle contained in S and denote the set consisting of indices of its vertices by Y_t;
            Find another edge in G if $|Y_t| = 2$; else find a triangle such that there exists a bijection f from X_t to Y_t and $\{e_{i,j}\}_{(i,j)\in X_t \times Y_t \setminus \{(i,f(i)):i\in X_t\}} \subseteq \bar{\mathcal{E}}$, where X_t denotes the set consisting of indices of its vertices;
            $\mathcal{E} \leftarrow \mathcal{E} \cup \{e_{i,j}\}_{(i,j)\in X_t \times Y_t \setminus \{(i,f(i)):i\in X_t\}}$;
            $X_{t;j} \leftarrow X_t \setminus \{f^{-1}(j)\}$, $Y_{t;i} \leftarrow Y_t \setminus \{f(i)\}$, $g_t \leftarrow g$, $l_{t;i} \leftarrow 0$, for $i \in X_t$, $j \in Y_t$;
            Add $C_t(X_t, Y_t, \{X_{t;j}\}_{j\in Y_t}, \{Y_{t;i}\}_{i\in X_t}, g_t, (l_{t;j})_{j\in Y_t})$ to $\mathcal{G}$;
            $t \leftarrow t + 1$;
        end for
        $g \leftarrow g + 1$;
    end for
end for

//Assign associated cooperation levels to $\mathcal{G}$, following the notation specified in Table 3.2

for $v_i \in V$ **do**
    Find the set A_i consisting of group numbers of those cycle groups that contain at least a cycle C_t with $i \in Y_t$;
    for $g \in A_i$ **do**
        Find the set $R_{i;g}$ consisting of all cycles C_t such that $g_t = g$ and $i \in Y_t$;
        Denote the average distance of all nodes v_j, $j \in X_{t;i}$, $t \in R_{i;g}$, by z_i;
    end for
    Order $(z_i)_{i\in[a(S)]}$ from the smallest to the largest, and obtain $(z_{\pi'(i)})_{i\in[a(S)]}$, where $\pi'(i)$ is a permutation of elements from $[a(S)]$;
    for $g \in A_i$ and $t \in R_{i;g}$ **do**
        $l_{t;j} \leftarrow \pi'(i) + 1$, for $j \in X_{t;i}$;
    end for
end for

Construction 3, the components $\mathbf{A}_{x,x}$, $\mathbf{U}_x$, and $\mathbf{B}_{x,i}$, $i \in [p] \setminus \{x\}$, are built locally. Suppose cloud $p+1$ is added into a double-level configuration adopting Construction 3. Algorithm 2 presents a procedure for adding this cloud, which only results in adding some columns and rows to the original generator matrix without changing the existing ones. Thus, the existing infrastructure does not need to be changed; each node only needs to add cross parities it receives from the newly added node to its current parities. Moreover, with this algorithm, the erasure correction capabilities $\{d_{1,i}\}_{v_i \in \mathcal{N}_{p+1}}$ of neighboring nodes of v_{p+1} are increased by δ_{p+1}.

The complexity of Algorithm 2 is roughly calculated as follows. Steps 1 and 2 need no calculations. Step 3 needs $O\left(\sum_{v_i \in \mathcal{N}_{p+1}} \delta_i k_{p+1}\right)$ operations at node v_{p+1}. Step 4 needs $O\left(\delta_{p+1} k_i\right)$ operations at node v_i, $v_i \in \mathcal{N}_{p+1}$. Step 5 needs $O\left(k_{p+1} r_{p+1} + |\mathcal{N}_{p+1}| \delta_{p+1} + \delta_{p+1} r_{p+1}\right)$ operations (since all $\mathbf{m}_i \mathbf{B}_{i,p+1}$ are computed in step 4, here we just need to compute the sum of them). Step 6 needs $O\left(\delta_i r_i\right)$ operations at node v_i, $v_i \in \mathcal{N}_{p+1}$ (since all $\mathbf{m}_{p+1} \mathbf{B}_{p+1,i}$ are computed in step 3). Therefore, node v_{p+1} needs $O\left(k_{p+1}(q - k_{p+1}) + |\mathcal{N}_{p+1}| \delta_{p+1} + \delta_{p+1} r_{p+1}\right)$ operations overall, and each node v_i needs $O(\delta_{p+1} n_i)$ operations overall, $v_i \in \mathcal{N}_{p+1}$.

Algorithm 2 Node Addition

Inputs:

$G(V, E)$: existing DSN;
p: number of nodes in $G(V, E)$;
v_{p+1}: the newly added node;
r_{p+1}: the message length of v_{p+1};
k_{p+1}: the number of parity symbols of the v_{p+1};
δ_{p+1}: the number of additional parities v_{p+1} provides globally to the DSN;
$\mathcal{N}_{p+1}$: the set of nodes to be connected to v_{p+1};
$\mathbf{G}$: the original generator matrix;

Outputs:

$\mathbf{G}$: the updated generator matrix;

1: Node v_{p+1} chooses its local parameters $\mathbf{A}_{p+1,p+1}$, $\mathbf{U}_{p+1}$, and $\mathbf{B}_{p+1,i}$, $v_i \in \mathcal{N}_{p+1}$;
2: Node v_i chooses additional cross parity matrices $\mathbf{B}_{i,p+1}$, $v_i \in \mathcal{N}_{p+1}$;
3: Node v_{p+1} sends $\mathbf{m}_{p+1} \mathbf{B}_{p+1,i}$ to node v_i, $v_i \in \mathcal{N}_{p+1}$;
4: Node v_i sends $\mathbf{m}_i \mathbf{B}_{i,p+1}$ to v_{p+1}, $v_i \in \mathcal{N}_{p+1}$;
5: Node v_{p+1} computes the stored codeword $\mathbf{c}_{p+1} = \mathbf{m}_{p+1} \mathbf{A}_{p+1,p+1} + \sum_{i \in \mathcal{N}} \mathbf{m}_i \mathbf{B}_{i,p+1} \mathbf{U}_{p+1}$;
6: Node v_i adds $\mathbf{m}_{p+1} \mathbf{B}_{p+1,i}$ to its current parity symbols, $v_i \in \mathcal{N}_{p+1}$;
7: Update $\mathbf{G}$ accordingly;

Example 14. *Consider again the set up in Example 3. Suppose a node v_{13} is to be added to the existing DSN and is to be connected to nodes v_9, v_{10}, and v_{12}, as shown in Fig. 3.20. The messages near the edges marked in red are sent from v_{13}, while those marked in blue are sent from the neighboring nodes v_9, v_{10}, and v_{12}, to v_{13}. Note that the new node v_{13} has coding parameters chosen independently from the existing nodes according to Theorem 2, which means that it naturally achieves scalability.*

Remark 5. *Note that provided the constraint on the lower bound of the field size at each node $q \geq (n_i + \delta_i')$ (see Construction 1), the Galois field might need an update if the existing field size no longer satisfies the requirement. There are two possible ways to update the Galois field.*

1. *Use the smallest extension field of the current field. This way requires recalculation of only few existing entries in the parity check matrices since the base field is a subfield of the extension field. However, the smallest extension field of $\mathrm{GF}(2^{\theta})$ is $\mathrm{GF}(2^{2\theta})$. Nonetheless, the scaling of the network is still efficient for the following reasons:*

   - *The quadratic field growth is an upper bound and only happens at the boundary, i.e., for any node addition such that the required field size is between 2^{θ} and $2^{2\theta}$, the field $\mathrm{GF}(2^{2\theta})$ always accommodates the network scaling.*
   - *In this case, it is not necessary for all the symbols at all the nodes to use the same Galois field. Instead, different nodes can use different fields provided that any two different fields used by two nodes are such that one is an extension field of the other. Moreover, the systematic part at each node can always remain unchanged. Therefore, not all the nodes need a larger field; at most only the new nodes and their neighboring nodes do.*

2. *Linearly scale the field to some predicted size. This way requires recalculation of all the existing entries, since the new field is not an extension field of the old one.*

The first way is more efficient when the current field size is relatively small and is typically applicable in sparse networks (which should be the case in practical applications). The second way is more efficient when the current field size approaches a prespecified upper bound on the network size, i.e., the required field size will never exceed some upper bound M, where M is large, while the current field size is closer to M than it is to $\sqrt{M}$.

3.5.3 Flexibility

The concept of flexibility was proposed and investigated for dynamic cloud storage in [22]. In a dynamic cloud storage system, the rate of which a given piece of data is accessed is likely to change. When the data stored at a cloud become hot, i.e., of higher demand, splitting the cloud into two smaller clouds effectively reduces the latency. However, this action should be done without reducing the erasure correction capability of the rest of the system or changing the remaining components.

Specifically, if the data stored at a cloud $v_i \in V$ become unexpectedly hot, the DSN needs to split v_i into two separate smaller clouds $v_{i^{\mathrm{a}}}$ and $v_{i^{\mathrm{b}}}$ to maintain relatively low latency; Algorithm 3 presents the procedure to do this. For simplicity, we focus here on the case where only the 1-st level cooperation is involved in, as presented in Construction 4.

The complexity of Algorithm 3 is calculated as follows. Steps 1–4 are direct splitting of the component matrices and no calculations are needed. Step 5 needs local decoding of $\mathbf{c}_i$ to obtain $\mathbf{s}_i$ and $\mathbf{m}_i$, which requires inverting a Cauchy matrix with size not exceeding $(r_i - \delta_i)$, which in turn typically requires $O\left((r_i - \delta_i)^2\right)$ operations provided the explicit formula of the inverse of a Cauchy matrix. There also exist methods that require $O\left((r_i - \delta_i)\left(\log(r_i - \delta_i)\right)^2\right)$ operations only to obtain the inverse. Step 6 needs $O\left(\delta_{i^{\mathrm{a}}}(k_{i^{\mathrm{b}}} + r_{i^{\mathrm{a}}})\right)$ operations. Therefore, the overall complexity is

$$O\left(\delta_{i^{\mathrm{a}}}(k_{i^{\mathrm{b}}} + r_{i^{\mathrm{a}}}) + (r_i - \delta_i)\left(\log(r_i - \delta_i)\right)^2\right).$$

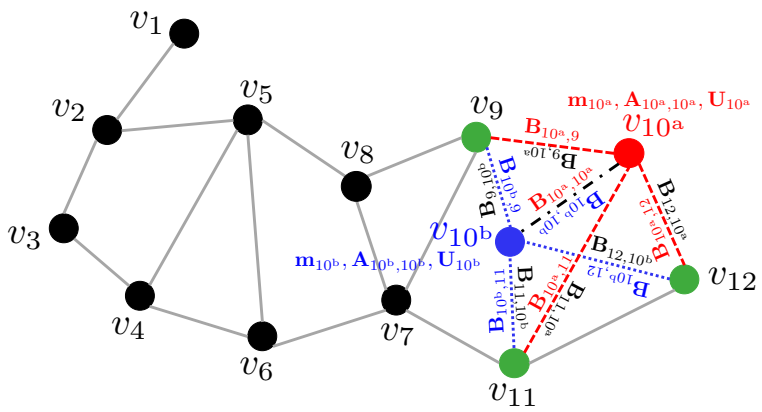


Figure 3.21: Flexibility: Split a node to two nodes in a DSN when it gets hot such that accessing the codeword stored at each one of them achieves low latency.

Note that the matrix $\mathbf{B}_{i,j}$ is vertically split into $\mathbf{B}_{i^a,j}$ and $\mathbf{B}_{i^b,j}$, while $\mathbf{B}_{j,i}$ is horizontally split into $\mathbf{B}_{j,i^a}$ and $\mathbf{B}_{j,i^b}$, for all v_j that are neighboring nodes of v_i. Therefore, it is obvious that $\mathbf{m}_i\mathbf{B}_{i,j} = \mathbf{m}_{i^a}\mathbf{B}_{i^a,j} + \mathbf{m}_{i^b}\mathbf{B}_{i^b,j}$ and one can prove that the local codeword $\mathbf{c}_j$ doesn't change for v_j that is a neighboring node of v_i. Moreover, since both the local and the global parity-check matrices for each non-split cloud remain unchanged, the local and global erasure capabilities of them are not affected according to Lemma 3. Furthermore, one can prove that the local codewords stored at the new clouds i^a and i^b tolerate $(r_{i^a} - \delta_{i^a})$ and $(r_{i^b} - \delta_{i^b})$ local erasures, respectively.

Example 15. *Consider again Example 3. If the data stored at node v_{10} become unexpectedly hot, then we split v_{10} into two separate nodes v_{10^a} and v_{10^b} following Algorithm 3, as shown in Fig. 3.21.*

Originally, node v_{10} needs to access all $n_{10} = k_{10} + r_{10}$ symbols to obtain message $\mathbf{m}_{10}$, which results in high latency when one has to access any set of symbols from $\mathbf{m}_{10}$ frequently. This operation results in unnecessary cost in terms of data processing times, which can be solved by splitting v_{10} into two nodes v_{10^a} and v_{10^b} that store $n_{10^a} = k_{10^a} + r_{10^a}$ and $n_{10^b} = k_{10^b} + r_{10^b}$ symbols, which contain the information of $\mathbf{m}_{10^a}$ and $\mathbf{m}_{10^b}$, respectively. Local access to each one of the two nodes will require significantly lower latency compared with a full access of the original node v_{10}. This approach improves the latency especially if

Algorithm 3 Node Splitting

Inputs:

$G(V, E)$: existing DSN;

v_i: the node to be split in $G(V, E)$;

$\mathcal{N}_i$: the set of neighboring nodes of v_i;

$v_{i^{\mathrm{a}}}$, $v_{i^{\mathrm{b}}}$: nodes v_i is split into;

k_i, $k_{i^{\mathrm{a}}}$, $k_{i^{\mathrm{b}}}$: the message lengths of v_i, $v_{i^{\mathrm{a}}}$, $v_{i^{\mathrm{b}}}$, respectively; $k_i = k_{i^{\mathrm{a}}} + k_{i^{\mathrm{b}}}$;

r_i, $r_{i^{\mathrm{a}}}$, $r_{i^{\mathrm{b}}}$: the number of parity symbols of v_i, $v_{i^{\mathrm{a}}}$, $v_{i^{\mathrm{b}}}$, respectively; $r_i = r_{i^{\mathrm{a}}} + r_{i^{\mathrm{b}}}$;

δ_i, $\delta_{i^{\mathrm{a}}}$, $\delta_{i^{\mathrm{b}}}$: the number of additional parities v_i, $v_{i^{\mathrm{a}}}$, $v_{i^{\mathrm{b}}}$ provides globally to the DSN, respectively; $\delta_i = \delta_{i^{\mathrm{a}}} + \delta_{i^{\mathrm{b}}}$;

$\mathbf{G}$: the original generator matrix;

Outputs:

$\mathbf{G}$: the updated generator matrix;

1: Node v_i splits $\mathbf{A}_{i,i}$ into $\mathbf{A}_{i^{\mathrm{a}},i^{\mathrm{a}}}$, $\mathbf{B}_{i^{\mathrm{b}},i^{\mathrm{a}}}$, $\mathbf{A}_{i^{\mathrm{b}},i^{\mathrm{b}}}$, $\mathbf{B}_{i^{\mathrm{a}},i^{\mathrm{b}}}$ as follows:
$\mathbf{A}_{i^{\mathrm{a}},i^{\mathrm{a}}} = \mathbf{A}_{i,i}\left[1 : k_{i^{\mathrm{a}}}, 1 : r_{i^{\mathrm{a}}}\right]$,
$\mathbf{B}_{i^{\mathrm{b}},i^{\mathrm{a}}} = \mathbf{A}_{i,i}\left[k_{i^{\mathrm{a}}} + 1 : k_i, 1 : \delta_{i^{\mathrm{a}}}\right]$,
$\mathbf{A}_{i^{\mathrm{b}},i^{\mathrm{b}}} = \mathbf{A}_{i,i}\left[k_{i^{\mathrm{a}}} + 1 : k_i, r_{i^{\mathrm{a}}} + 1 : r_i\right]$,
$\mathbf{B}_{i^{\mathrm{a}},i^{\mathrm{b}}} = \mathbf{A}_{i,i}\left[1 : k_{i^{\mathrm{a}}}, r_{i^{\mathrm{a}}} + 1 : r_{i^{\mathrm{a}}} + \delta_{i^{\mathrm{b}}}\right]$;

2: Node v_i splits $\mathbf{B}_{i,j}$, $\forall v_j \in \mathcal{N}_i$, into $\mathbf{B}_{i^{\mathrm{a}},j}$ and $\mathbf{B}_{i^{\mathrm{b}},j}$ as follows:
$\mathbf{B}_{i^{\mathrm{a}},j} = \mathbf{B}_{i,j}\left[1 : k_{i^{\mathrm{a}}}, 1 : \delta_j\right]$,
$\mathbf{B}_{i^{\mathrm{b}},j} = \mathbf{B}_{i,j}\left[k_{i^{\mathrm{a}}} + 1 : k_i, 1 : \delta_j\right]$;

3: Node v_j, $\forall v_j \in \mathcal{N}_i$, splits $\mathbf{B}_{j,i}$ into $\mathbf{B}_{j,i^{\mathrm{a}}}$ and $\mathbf{B}_{j,i^{\mathrm{b}}}$ as follows:
$\mathbf{B}_{j,i^{\mathrm{a}}} = \mathbf{B}_{j,i}\left[1 : k_j, 1 : \delta_{i^{\mathrm{a}}}\right]$,
$\mathbf{B}_{j,i^{\mathrm{b}}} = \mathbf{B}_{j,i}\left[1 : k_j, \delta_{i^{\mathrm{a}}} + 1 : \delta_i\right]$, $\forall v_j \in \mathcal{N}_i$;

4: Node v_i splits $\mathbf{U}_i$ into $\mathbf{U}_{i^{\mathrm{a}}}$ and $\mathbf{U}_i^{\mathrm{b}}$ as follows:
$\mathbf{U}_{i^{\mathrm{a}}} = \mathbf{U}_i\left[1 : \delta_{i^{\mathrm{a}}}, 1 : r_{i^{\mathrm{a}}}\right]$,
$\mathbf{U}_{i^{\mathrm{b}}} = \mathbf{U}_i\left[\delta_{i^{\mathrm{a}}} + 1 : \delta_1, r_{i^{\mathrm{a}}} + 1 : r_{i^{\mathrm{b}}}\right]$;

5: Compute the additional cross parities $\mathbf{s}_i$'s by solving the equation $\mathbf{s}_i\mathbf{U}_i = \mathbf{c}_i - \mathbf{m}_i\mathbf{A}_{i,i}$, where $i \in [p]$. Find $\mathbf{s}_{i^{\mathrm{a}}} \in \mathrm{GF}(q)^{\delta_{i^{\mathrm{a}}}}$, $\mathbf{s}_{i^{\mathrm{b}}} \in \mathrm{GF}(q)^{\delta_{1^{\mathrm{b}}}}$ such that $\mathbf{s}_i = [\mathbf{s}_{i^{\mathrm{a}}}, \mathbf{s}_{i^{\mathrm{b}}}]$;

6: Compute the message stored at the node $v_{i^{\mathrm{a}}}$ and $v_{i^{\mathrm{a}}}$ as follows:
$\mathbf{c}_{i^{\mathrm{a}}} = \left[\mathbf{m}_{i^{\mathrm{a}}}, \mathbf{m}_{i^{\mathrm{a}}}\mathbf{A}_{i^{\mathrm{a}},i^{\mathrm{a}}} + \left(\mathbf{m}_{i^{\mathrm{b}}}\mathbf{B}_{i^{\mathrm{b}},i^{\mathrm{a}}} + \mathbf{y}_{i^{\mathrm{a}}}\right)\mathbf{U}_{i^{\mathrm{a}}}\right]$,
$\mathbf{c}_{i^{\mathrm{b}}} = \left[\mathbf{m}_{i^{\mathrm{b}}}, \mathbf{m}_{i^{\mathrm{b}}}\mathbf{A}_{i^{\mathrm{b}},i^{\mathrm{b}}} + \left(\mathbf{m}_{i^{\mathrm{a}}}\mathbf{B}_{i^{\mathrm{a}},i^{\mathrm{b}}} + \mathbf{y}_{i^{\mathrm{b}}}\right)\mathbf{U}_{i^{\mathrm{b}}}\right]$;

7: Update $\mathbf{G}$ accordingly;

the erasures are bursty, i.e., concentrated within any one of $\mathbf{c}_{10^a}$ *or* $\mathbf{c}_{10^b}$*. Even if the erasures are distributed more evenly among* $\mathbf{c}_{10^a}$ *and* $\mathbf{c}_{10^b}$*, the total processing time to obtain* $\mathbf{m}_{10}$ *will be the maximum of their individual processing times, which is still much shorter than the original time.*

3.6 Conclusion

Hierarchical locally accessible codes in the context of centralized cloud networks have been discussed in various prior works, whereas those of DSNs (no prespecified topology) have not been explored. In this chapter, we proposed a topology-adaptive cooperative data protection scheme for DSNs, which significantly extends our previous work on hierarchical coding for centralized distributed storage. We discussed the recoverable erasure patterns of our proposed scheme, demonstrating that our scheme corrects patterns pertaining to dynamic DSNs. Our scheme achieves faster recovery speed compared with existing network coding methods, and enables an intrinsic information flow from nodes with higher reliability to nodes with lower reliability that are close to them on the network. Moreover, our constructions are also proved to be scalable and flexible, making them a construction with great potential to be employed in dynamic DSNs.

CHAPTER 4

Spatially-Coupled Codes with High Memories

4.1 Introduction

Spatially-coupled (SC) codes, also known as low-density parity-check (LDPC) codes with convolutional structures, are an ideal choice for streaming applications and data storage devices thanks to their threshold saturation phenomenon [37, 38, 39, 3, 4] and amenability to low-latency windowed decoding [69]. SC codes are constructed by partitioning the parity-check matrix of an underlying block code, followed by rearranging the component matrices in a *convolutional* manner. In particular, component matrices are vertically concatenated into a *replica*, and then multiple replicas are horizontally placed together, resulting in a *coupled* code. The number of component matrices minus one is referred to as the *memory* of the SC code [70, 40, 41, 71].

It is known that the performance of an SC code improves as its memory increases. This is a byproduct of improved node expansion and additional degrees of freedom that can be utilized to decrease the number of short cycles and detrimental objects [40, 41, 42, 43, 44]. A plethora of existing works [40, 41, 45, 47] focus on minimizing the number of short cycles in the graph of the SC code. Although the optimization problem of designing SC codes with memory less than 4 has been efficiently solved [40, 41], there is still an absence of efficient algorithms that construct good enough SC codes with high memories systematically. Esfahanizadeh *et al.* [40] proposed a combinatorial framework to develop optimal quasi-cyclic (QC)

SC codes, comprising so-called optimal overlap (OO) to search for the optimal partitioning matrices, and circulant power optimization (CPO) to optimize the lifting parameters, which was extended by Hareedy *et al.* [41]. However, this method is hard to execute in practice for high-memory codes due to the increasing computational complexity. Heuristic methods that search for good SC codes with high memories are derived in [45, 46, 47, 44]. However, high-memory codes designed by purely heuristic methods are unable to reach the potential performance gain that can be achieved through high memories due to lack of theoretical properties; several of these codes can even be beat by optimally designed QC-SC codes with lower memories under the same constraint length [47]. Therefore, a method that theoretically identifies an avenue to a near-optimal construction of SC codes with high memories is of significant interest.

Inspired by the excellent performance and the low computational complexity offered by approaches comprising theoretical analysis guiding heuristic methods, we propose a two-step hybrid optimization framework that has these advantages. The framework first specifies a search subspace that is theoretically proved to be locally optimal, followed by a semi-greedy algorithm within this targeted search space. By analogy with threshold optimization approaches that search for LDPC ensembles with the optimal degree distribution, our first step is to obtain an SC ensemble with the optimal edge distribution (i.e., density distribution of component matrices). The associated metric is the expected number of targeted detrimental objects in the protograph of the code. Having reached a locally optimal edge distribution through gradient descent, we then apply a semi-greedy algorithm to search for a locally optimal partitioning matrix that satisfies this edge distribution. Our probabilistic framework is referred to as **gradient-descent distributor, algorithmic optimizer (GRADE-AO)**.

Preliminary version of this work was presented in [57], where we focused only on the minimization of the number of short cycles. In this work, we develop a general framework that handles arbitrary objects. While cycles are detrimental in codes with low variable node

(VN) degrees, such as regular codes with VN degree 2 or 3, objects that dominate the error profiles in higher-degree codes and irregular codes are typically more advanced. In particular, we focus on the concatenation of two short cycles in this chapter. These concatenated cycles are common subgraphs of the detrimental objects, which are absorbing sets (ASs) [42]), that govern the performance of LDPC codes with VN degree ≥ 3 in error floor region. These detrimental objects are also the major source of undesirable dependencies that undermine the performance in the waterfall region. While focusing on cycles for simplicity, which is the case for the majority of existing works, unnecessary degrees of freedom could be exhausted on isolated cycles that are much less problematic. To the best of our knowledge, we are the first to provide a framework that systematically eliminates objects other than cycles from the Tanner graph of an SC code with mathematical guarantees, which is important for a variety of applications including storage systems. Hareedy *et al.* [72, 73] proposed the so-called weight consistency matrix (WCM) framework to search for edge-weight assignments that minimize the number of ASs in non-binary (NB) LDPC codes with a given underlying topology, and demonstrated performance gains in data storage systems. Simulation results show that our framework leads to codes with excellent performance in flash memory and magnetic recording systems. The proposed GRADE-AO framework not only opens a door to fine-grained optimization over detrimental objects in SC codes, but also can be applied in optimizing the unweighted graphs of non-binary (NB) SC codes, which can lead to excellent NB-SC codes when combined with the WCM framework. Because of the improved threshold and waterfall performance, GRADE-AO has potential to produce SC codes for communication systems as well.

In this chapter, we propose a probabilistic framework that efficiently searches for near-optimal SC codes with high memories. In Section 4.2, we introduce preliminaries of SC codes and the performance-related metrics. In Section 4.3, we develop the theoretical basis of GRADE, which derives an edge distribution that determines a locally optimal SC ensemble. In Section 4.4, we introduce the theoretical details of how GRADE is generalized to

more sophisticated objects. The distribution obtained through GRADE leads to effective initialization and specifies the search space of the semi-greedy algorithm adopted in AO afterwards. In Section 4.5, we introduce two examples of GRADE-AO that result in near-optimal SC codes: the so-called **gradient-descent (GD) codes** and **topologically-coupled (TC) codes**. In summary, we generalize GRADE to a rich class of relevant objects and present examples of GRADE-AO that focus on concatenated cycles. Our proposed framework is supported in Section 4.6 by simulation results of seven groups of codes, with the best code in each obtained from GRADE-AO. Finally, we make concluding remarks and introduce possible future work in Section 4.7.

4.2 Preliminaries

In this section, we recall the typical construction of SC codes with quasi-cyclic (QC) structure. Any QC code with a parity-check matrix $\mathbf{H}$ is obtained by replacing each nonzero (zero) entry of some binary matrix $\mathbf{H}^{\mathrm{P}}$ with a circulant (zero) matrix of size z, $z \in \mathbb{N}$. The matrix $\mathbf{H}^{\mathrm{P}}$ and z are referred to as the protograph and the circulant size of the code, respectively. In particular, the protograph $\mathbf{H}_{\mathrm{SC}}^{\mathrm{P}}$ of an SC code has a convolutional structure composed of L replicas, as presented in Fig. 4.1. Each replica is obtained by stacking the disjoint component matrices $\{\mathbf{H}_i^{\mathrm{P}}\}_{i=0}^{m}$, where m is the memory and $\mathbf{\Pi} = \mathbf{H}_0^{\mathrm{P}} + \mathbf{H}_1^{\mathrm{P}} + \cdots + \mathbf{H}_m^{\mathrm{P}}$ is the protograph of the underlying block code.

In this chapter, we constrain $\mathbf{\Pi}$ to be an all-one matrix of size $\gamma \times \kappa$, $\gamma, \kappa \in \mathbb{N}$. An SC code is then uniquely represented by its partitioning matrix $\mathbf{P}$ and lifting matrix $\mathbf{L}$, where $\mathbf{P}$ and $\mathbf{L}$ are all $\gamma \times \kappa$ matrices. The matrix $\mathbf{P}$ has $(\mathbf{P})_{i,j} = a$ if $(\mathbf{H}_a^{\mathrm{P}})_{i,j} = 1$. The matrix $\mathbf{L}$ is determined by replacing each circulant matrix by its associated exponent. Here, this exponent represents the power to which the matrix $\boldsymbol{\sigma}$ defined by $(\boldsymbol{\sigma})_{i,i+1} = 1$ is raised, where $(\boldsymbol{\sigma})_{z,z+1} = (\boldsymbol{\sigma})_{z,1}$.

The performance of finite-length LDPC codes is strongly affected by the number of detrimental objects that are subgraphs with certain structures in the Tanner graphs of those

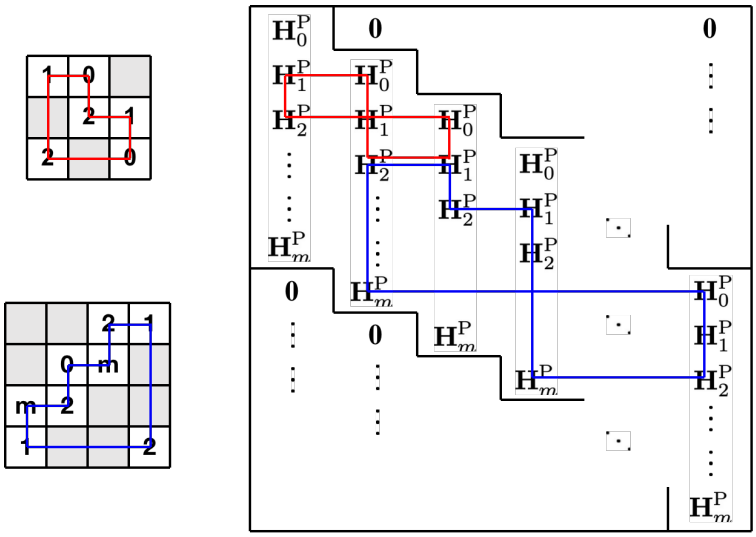


Figure 4.1: Cycles in the protograph (right panel) and their corresponding structures in the partitioning matrices (left panel).

codes. Two major classes of detrimental objects are trapping sets and absorbing sets. Since enumerating and minimizing the number of detrimental objects is complicated, existing work typically focuses on common substructures of these objects: the short cycles [40, 41, 45]. A cycle-$2g$ candidate in $\mathbf{H}_{\mathrm{SC}}^{\mathrm{P}}$ ($\mathbf{\Pi}$) is a path of traversing a structure to generate cycles of length $2g$ after lifting (partitioning) [41]. In an SC code, each cycle in the Tanner graph corresponds to a cycle candidate in the protograph $\mathbf{H}_{\mathrm{SC}}^{\mathrm{P}}$, and each cycle candidate in $\mathbf{H}_{\mathrm{SC}}^{\mathrm{P}}$ corresponds to a cycle candidate C in the base matrix $\mathbf{\Pi}$. Lemma 4 specifies a necessary and sufficient condition for a cycle candidate in $\mathbf{\Pi}$ to become a cycle candidate in the protograph and then a cycle in the final Tanner graph.

Lemma 4. *Let C be a cycle-$2g$ candidate in the base matrix, where $g \in \mathbb{N}$, $g \geq 2$. Denote C by $(j_1, i_1, j_2, i_2, \ldots, j_g, i_g)$, where (i_k, j_k), (i_k, j_{k+1}), $1 \leq k \leq g$, $j_{g+1} = j_1$, are nodes of C in $\mathbf{\Pi}$, $\mathbf{P}$, and $\mathbf{L}$. Then C becomes a cycle candidate in the protograph if and only if the following condition follows [45]:*

$$\sum_{k=1}^{g} \mathbf{P}(i_k, j_k) = \sum_{k=1}^{g} \mathbf{P}(i_k, j_{k+1}). \tag{4.1}$$

This cycle candidate becomes a cycle in the Tanner graph if and only if [74]:

$$\sum_{k=1}^{g} \mathbf{L}(i_k, j_k) \equiv \sum_{k=1}^{g} \mathbf{L}(i_k, j_{k+1}) \mod z. \tag{4.2}$$

As shown in Fig. 4.1, a cycle-6 candidate and a cycle-8 candidate in the partitioning matrix with assignments satisfying condition (5.9), and their corresponding cycle candidates in the protograph are marked by red and blue, respectively. An optimization of a QC-SC code is typically divided into two major steps: optimizing $\mathbf{P}$ to minimize the number of cycle candidates in the protograph, and optimizing $\mathbf{L}$ to further reduce that number in the Tanner graph given the optimized $\mathbf{P}$ [40, 41]. The latter goal has been achieved in [40] and [41], using an algorithmic method called circulant power optimization (CPO), while the former goal is yet to be achieved for large m. We note that the step separation highlighted above notably reduces the overall optimization complexity.

In the remainder of this chapter, we first focus on QC-SC codes for the additive white Gaussian noise (AWGN) channel, where the most detrimental objects are the low weight absorbing sets (ASs) [40]. The ASs are defined in Definition 9.

Definition 9. ***(Absorbing Sets)*** *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 $\mathrm{GF}(q)\backslash\{0\}$ and set all other VNs to 0. The set $\mathcal{V}$ is said to be an (a, b)* ***absorbing set (AS)*** *over $\mathrm{GF}(q)$ if the size of $\mathcal{V}$ is a, the number of unsatisfied neighboring CNs of $\mathcal{V}$ is b, and each VN in $\mathcal{V}$ is connected to strictly more satisfied than unsatisfied neighboring CNs, for some set of VN values.*

An (a, b) ***elementary AS*** *$\mathcal{V}$ over $\mathrm{GF}(q)$ is an (a, b) AS with the additional property that all the satisfied (resp., unsatisfied (if any)) neighboring CNs of $\mathcal{V}$ have degree 2 (resp., degree 1); otherwise the AS is referred to as an (a, b)* ***non-elementary AS****.*

Consider a subgraph induced by a subset $\mathcal{V}$ of VNs in the Tanner graph of a binary code. The set $\mathcal{V}$ is said to be an (a, b) ***binary AS*** *if the size of $\mathcal{V}$ is a, the number of odd-degree neighboring CNs of $\mathcal{V}$ is b, and each VN in $\mathcal{V}$ is connected to strictly more even-degree than*

odd-degree neighboring CNs.

Observe that the unlabeled configuration (all edge weights set to 1*) underlying an* (a, b) *non-binary elementary AS is itself an* (a, b) *binary elementary AS.*

Consequently, a simplified optimization focuses on cycle candidates of lengths 4, 6, and 8 [40, 41]. Existing literature shows that the optimal $\mathbf{P}$ for an SC code with $m \leq 2$ typically has a balanced (uniform) edge distribution among component matrices [40]. However, in the remaining sections, we show that the edge distribution for optimal SC codes with large m is not uniform, and we propose the GRADE-AO framework that explores a locally optimal solution. With the success in cycle optimization, we step forward to a finer-grained optimization over more advanced objects, which can be applied in higher degree codes and irregular codes. While GRADE can be generalized for arbitrary objects, we present constructions obtained though GRADE-AO focusing on concatenated cycles. Simulation results show that our proposed codes have excellent performance on practical channel models derived from flash memories and magnetic recording (MR), in both waterfall and error floor region.

4.3 A Probabilistic Optimization Framework

In this section, we present a probabilistic framework that searches for a locally optimal edge distribution for the partitioning matrices of SC codes with given memories through the gradient-descent algorithm.

Definition 10. *Let* $\gamma, \kappa, m, m_t \in \mathbb{N}$ *and* $\mathbf{a} = (a_0, a_1, \ldots, a_{m_t})$*, where* $0 = a_0 < a_1 < \cdots < a_{m_t} = m$*. A* (γ, κ) *SC code with memory* m *is said to have* ***coupling pattern*** $\mathbf{a}$ *if and only if* $\mathbf{H}_i^{\mathrm{P}} \neq \mathbf{0}^{\gamma \times \kappa}$*, for all* $i \in \{a_0, a_1, \ldots, a_{m_t}\}$*, and* $\mathbf{H}_i^{\mathrm{P}} = \mathbf{0}^{\gamma \times \kappa}$*, otherwise. The value* m_t *is called the* ***pseudo-memory*** *of the SC code.*

4.3.1 Probabilistic Metric

In this subsection, we define metrics relating the edge distribution to the expected number of cycle candidates in the protograph in Theorem 6 and Theorem 7. While Schmalen *et al.* have shown in [75] that nonuniform coupling (nonuniform edge distribution in this chapter) yields an improved threshold, our work differs in two areas: 1) Explicit optimal coupling graphs were exhaustively searched and were restricted to small memories in [75], whereas our method produces near-optimal SC protographs for arbitrary memories. 2) Work [75] focused on the asymptotic analysis for the threshold region, while our framework is dedicated to the finite-length construction and has additional demonstrable gains in the error floor region.

Definition 11. *Let* $m, m_t \in \mathbb{N}$ *and* $\mathbf{a} = (a_0, a_1, \ldots, a_{m_t})$*, where* $0 = a_0 < a_1 < \cdots < a_{m_t} = m$*. Let* $\mathbf{p} = (p_0, p_1 \ldots, p_{m_t})$*, where* $0 < p_i \leq 1$*,* $p_0 + p_1 + \cdots + p_{m_t} = 1$*: each* p_i *specifies the probability of a '*1*' in* $\mathbf{\Pi}$ *going to the component matrix* $\mathbf{H}_{a_i}^{\mathrm{P}}$*, thus* $\mathbf{p}$ *is referred to as* ***edge distribution*** *under random partition later on. Then, the following* $f(X; \mathbf{a}, \mathbf{p})$*, which is abbreviated to* $f(X)$ *when the context is clear, is called the* ***coupling polynomial*** *of an SC code with coupling pattern* $\mathbf{a}$*, associated with probability distribution* $\mathbf{p}$*:*

$$f(X; \mathbf{a}, \mathbf{p}) \triangleq \sum\nolimits_{0 \leq i \leq m_t} p_i X^{a_i}. \tag{4.3}$$

Theorem 6. *Let* $[\cdot]_i$ *denote the coefficient of* X^i *of a polynomial. Denote by* $P_6(\mathbf{a}, \mathbf{p})$ *the probability of a cycle-*6 *candidate in the base matrix becoming a cycle-*6 *candidate in the protograph under random partitioning with edge distribution* $\mathbf{p}$*. Then,*

$$P_6(\mathbf{a}, \mathbf{p}) = \left[f^3(X) f^3(X^{-1}) \right]_0. \tag{4.4}$$

Proof. According to Lemma 4, suppose the cycle-6 candidate in the base matrix is repre-

sented by $C(j_1, i_1, j_2, i_2, j_3, i_3)$. Then,

$$\begin{aligned}
&P_6(\mathbf{a}, \mathbf{p}) = \mathbb{P}\left[\sum_{k=1}^{3} \mathbf{P}(i_k, j_k) = \sum_{k=1}^{3} \mathbf{P}(i_k, j_{k+1})\right] \\
&= \sum_{\sum_{k=1}^{3} x_k = \sum_{k=1}^{3} y_k} \prod_{k=1}^{3} \mathbb{P}\left[\mathbf{P}(i_k, j_k) = x_k, \mathbf{P}(i_k, j_{k+1}) = y_k\right] \\
&= \sum_{\sum_{k=1}^{3} x_k = \sum_{k=1}^{3} y_k} p_{x_1} p_{x_2} p_{x_3} p_{y_1} p_{y_2} p_{y_3} \\
&= \left[\sum_{x_k, y_k \in \mathrm{vals}(\mathbf{a})} p_{x_1} p_{x_2} p_{x_3} p_{y_1} p_{y_2} p_{y_3} X^{x_1 + x_2 + x_3 - y_1 - y_2 - y_3} \right]_0 \\
&= \left[f^3(X) f^3(X^{-1}) \right]_0,
\end{aligned}$$

where $\mathrm{vals}(\mathbf{a})$ is the set $\{a_0, a_1, \ldots, a_{m_t}\}$. Thus, the theorem is proved. □

Example 16. *Consider SC codes with full memories and uniform partition, i.e.,* $\mathbf{a} = (0, 1, \ldots, m)$ *and* $\mathbf{p} = \frac{1}{m+1}\mathbf{1}_{m+1}$. *When* $m = 2$, $P_6(\mathbf{a}, \mathbf{p}) = 0.1934$*; when* $m = 4$, $P_6(\mathbf{a}, \mathbf{p}) = 0.1121$.

Example 17. *First, consider SC codes with* $m = m_t = 2$. *Let* $\mathbf{a}_1 = (0, 1, 2)$ *and* $\mathbf{p}_1 = (2/5, 1/5, 2/5)$. *According to Theorem 6,* $f(X) = (2 + X + 2X^2)/5$, $f^3(X)f^3(X^{-1}) = 0.0041(X^6 + X^{-6}) + 0.0123(X^5 + X^{-5}) + 0.0399(X^4 + X^{-4}) + 0.0717(X^3 + X^{-3}) + 0.1267(X^2 + X^{-2}) + 0.1544(X + X^{-1}) + 0.1818$. *Therefore,* $P_6(\mathbf{a}_1, \mathbf{p}_1) = 0.1818$. *Second, consider SC codes with* $m = m_t = 4$. *Let* $\mathbf{a}_2 = (0, 1, 2, 3, 4)$ *and* $\mathbf{p}_2 = (0.31, 0.13, 0.12, 0.13, 0.31)$. *According to Theorem 6,* $P_6(\mathbf{a}_2, \mathbf{p}_2) = 0.0986$.

After we have derived the metric for cycle-6 candidates in the protograph, we now turn to the case of cycle-8 candidates. As shown in Fig. 4.2, cycle candidates in the base matrix that result in cycle-8 candidates in the protograph can be categorized into 6 different structures, labeled $S_1, \ldots, S_6$. Different cases are differentiated by the number of rows and columns (without order) the structures span in the partitioning matrix [41]. Specifically, $S_1, \ldots, S_6$ denote the structures that span submatrices of size 2×2, 2×3 or 3×2, 3×3, 2×4 or 4×2, 3×4 or 4×3, and 4×4, respectively. Any structure that belongs to S_2, S_4, S_5 has multiple

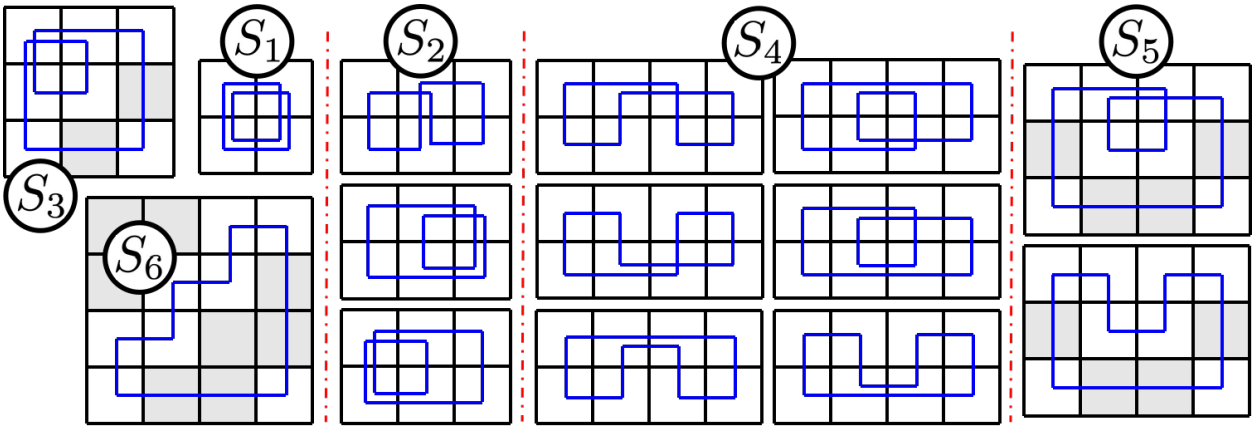


Figure 4.2: Structures and cycle candidates for cycle-8.

cycle-8 candidates, and these distinct candidates are marked by blue in Fig. 4.2.

Lemma 5. *Let $P_{8;i}(\mathbf{a}, \mathbf{p})$, $1 \leq i \leq 6$, denote the probability of a cycle-8 candidate of structure S_i in the base matrix becoming a cycle-8 candidate in the protograph, under random partition with edge distribution $\mathbf{p}$. Then,*

$$
\begin{aligned}
P_{8;1}(\mathbf{a}, \mathbf{p}) &= \left[f^2(X) f^2(X^{-1})\right]_0, \\
P_{8;2}(\mathbf{a}, \mathbf{p}) &= \left[f(X^2) f(X^{-2}) f^2(X) f^2(X^{-1})\right]_0, \\
P_{8;3}(\mathbf{a}, \mathbf{p}) &= \left[f(X^2) f^2(X) f^4(X^{-1})\right]_0, \text{ and} \\
P_{8;4}(\mathbf{a}, \mathbf{p}) &= P_{8;5}(\mathbf{a}, \mathbf{p}) = P_{8;6}(\mathbf{a}, \mathbf{p}) = \left[f^4(X) f^4(X^{-1})\right]_0.
\end{aligned}
$$

Proof. For structures where the nodes of the cycle-8 candidates are pairwise different, namely, S_4, S_5, S_6, the result can be derived by following the logic in the proof of Theorem 6.

For S_1, suppose the indices of the rows and columns are i_1, i_2, and j_1, j_2, respectively. Then, the cycle condition in Lemma 4 is $\mathbf{P}(i_1, j_1) + \mathbf{P}(i_2, j_2) = \mathbf{P}(i_1, j_2) + \mathbf{P}(i_2, j_1)$.

For S_2, suppose the indices of the rows and columns are i_1, i_2, and j_1, j_2, j_3, respectively. Then, the cycle condition in Lemma 4 is $2\mathbf{P}(i_1, j_1) - 2\mathbf{P}(i_2, j_1) + \mathbf{P}(i_2, j_2) + \mathbf{P}(i_2, j_3) - \mathbf{P}(i_1, j_2) - \mathbf{P}(i_1, j_3) = 0$.

For S_3, suppose the indices of the rows and columns are i_1, i_2, i_3, and j_1, j_2, j_3, respectively.

Then, the cycle condition in Lemma 4 is $2\mathbf{P}(i_1,j_1)+\mathbf{P}(i_2,j_2)+\mathbf{P}(i_3,j_3)-\mathbf{P}(i_1,j_2)-\mathbf{P}(i_2,j_1)-\mathbf{P}(i_1,j_3)-\mathbf{P}(i_3,j_1)=0$.

Following the logic in the proof of Theorem 6, the case for S_1, S_2, S_3 can be proved. □

Theorem 7. *Denote $N_8(\mathbf{a},\mathbf{p})$ as the expectation of the number of cycle-8 candidates in the protograph. Then,*

$$\begin{aligned} N_8(\mathbf{a},\mathbf{p}) = w_1\left[f^2(X)f^2(X^{-1})\right]_0 + w_2\left[f(X^2)f(X^{-2})f^2(X)f^2(X^{-1})\right]_0 \\ + w_3\left[f(X^2)f^2(X)f^4(X^{-1})\right]_0 + w_4\left[f^4(X)f^4(X^{-1})\right]_0, \end{aligned} \tag{4.5}$$

where $w_1 = \binom{\gamma}{2}\binom{\kappa}{2}$, $w_2 = 3\binom{\gamma}{2}\binom{\kappa}{3} + 3\binom{\gamma}{3}\binom{\kappa}{2}$, $w_3 = 18\binom{\gamma}{3}\binom{\kappa}{3}$, $w_4 = 6\binom{\gamma}{2}\binom{\kappa}{4} + 6\binom{\gamma}{4}\binom{\kappa}{2} + 36\binom{\gamma}{3}\binom{\kappa}{4} + 36\binom{\gamma}{4}\binom{\kappa}{3} + 24\binom{\gamma}{4}\binom{\kappa}{4}$.

Proof. Provided the results in Lemma 5, we just need to prove that the numbers of cycle candidates of structures $S_1, S_2, \ldots, S_6$ in a $\gamma \times \kappa$ base matrix are $\binom{\gamma}{2}\binom{\kappa}{2}$, $3\binom{\gamma}{2}\binom{\kappa}{3} + 3\binom{\gamma}{3}\binom{\kappa}{2}$, $18\binom{\gamma}{3}\binom{\kappa}{3}$, $6\binom{\gamma}{2}\binom{\kappa}{4} + 6\binom{\gamma}{4}\binom{\kappa}{2}$, $36\binom{\gamma}{3}\binom{\kappa}{4} + 36\binom{\gamma}{4}\binom{\kappa}{3}$, and $24\binom{\gamma}{4}\binom{\kappa}{4}$, respectively.

Take $i = 5$ as an example. The number of cycle candidates of structure S_5 in any 3×4 or 4×3 matrix is $3 \cdot \binom{4}{2} \cdot 2 = 36$. The total number of 3×4 or 4×3 matrices in a $\gamma \times \kappa$ base matrix is $\binom{\gamma}{3}\binom{\kappa}{4} + \binom{\gamma}{4}\binom{\kappa}{3}$. Therefore, the total number of cycle candidates of structure S_5 is $36\binom{\gamma}{3}\binom{\kappa}{4} + 36\binom{\gamma}{4}\binom{\kappa}{3}$. By a similar logic, we can prove the result for the remaining structures. □

Remark 6. *Note that each cycle candidate satisfying the cycle condition in the partitioning matrix can result in multiple cycle candidates in the protograph; the multiplicity is determined by its width, i.e., the number of replicas each resultant cycle candidate spans in the protograph. We ignore the number of replicas a cycle candidate spans in $\mathbf{H}_{\mathrm{SC}}^{\mathrm{P}}$. We address this number in the CPO stage.*

4.3.2 Gradient-Descent Distributor

By contrasting Examples 16 and 17 it is clear that for a given coupling pattern, an optimal edge distribution is not necessarily reached by a uniform partition. In this subsection, we develop an algorithm that obtains a locally optimal distribution by gradient descent.

Lemma 6. *Given $m_t \in \mathbb{N}$ and $\mathbf{a} = (a_0, a_1, \ldots, a_{m_t})$, a necessary condition for $P_6(\mathbf{a}, \mathbf{p})$ to reach its minimum value is that the following equation holds for some $c_0 \in \mathbb{R}$:*

$$\left[f^3(X)f^2(X^{-1})\right]_{a_i} = c_0,\ \forall i, 0 \le i \le m_t. \tag{4.6}$$

Proof. Consider the gradient of $L_6(\mathbf{a}, \mathbf{p}) = P_6(\mathbf{a}, \mathbf{p}) + c(1 - p_0 - p_1 - \cdots - p_{m_t})$.

$$\begin{aligned}
&\nabla_{\mathbf{p}} L_6(\mathbf{a}, \mathbf{p}) \\
=&\nabla_{\mathbf{p}} \left(P_6(\mathbf{a}, \mathbf{p}) + c(1 - p_0 - p_1 - \cdots - p_{m_t})\right) \\
=&\nabla_{\mathbf{p}} \left[f^3(X)f^3(X^{-1})\right]_0 - c\mathbf{1}_{m_t+1} \\
=&\left[\nabla_{\mathbf{p}} \left(f^3(X)f^3(X^{-1})\right)\right]_0 - c\mathbf{1}_{m_t+1} \\
=&3\left[f^2(X)f^2(X^{-1})f(X)\nabla_{\mathbf{p}} f(X^{-1})\right]_0 + 3\left[f^2(X)f^2(X^{-1})f(X^{-1})\nabla_{\mathbf{p}} f(X)\right]_0 - c\mathbf{1}_{m_t+1} \\
=&6\left[f^3(X)f^2(X^{-1})\left(X^{-a_0}, X^{-a_1}, \ldots, X^{-a_{m_t}}\right)\right]_0 - c\mathbf{1}_{m_t+1}.
\end{aligned} \tag{4.7}$$

When $P_8(\mathbf{a}, \mathbf{p})$ reaches its minimum, $\nabla_{\mathbf{p}} [L(\mathbf{a}, \mathbf{p})] = \mathbf{0}_{m_t+1}$, which is equivalent to (4.6) by defining $c_0 = c/6$. □

Lemma 7. *Given $\gamma, \kappa, m_t \in \mathbb{N}$ and $\mathbf{a} = (a_0, a_1, \ldots, a_{m_t})$, a necessary condition for $N_8(\mathbf{a}, \mathbf{p})$ to reach its minimum value is that the following equation holds for some $c_0 \in \mathbb{R}$:*

$$\begin{aligned}
&\left[4f^2(X)f(X^{-1})\right])_{a_i} + \bar{w}_2\left[2f(X^2)f^2(X)f^2(X^{-1})\right]_{2a_i} + \bar{w}_2\left[4f(X^2)f(X^{-2})f^2(X)f(X^{-1})\right]_{a_i} \\
+&\bar{w}_3\left[f^2(X)f^4(X^{-1})\right]_{-2a_i} + \bar{w}_3\left[2f(X^2)f(X)f^4(X^{-1})\right]_{-a_i} + \bar{w}_3\left[4f(X^2)f^2(X)f^3(X^{-1})\right]_{a_i} \\
+&\bar{w}_4\left[8f^4(X)f^3(X^{-1})\right]_{a_i} = c_0,\ \forall i, 0 \le i \le m_t,
\end{aligned}$$

where $\bar{w}_2 = \gamma + \kappa - 4$, $\bar{w}_3 = 2(\gamma-2)(\kappa-2)$, *and* $\bar{w}_4 = \frac{1}{2}\left[(\gamma-2)(\gamma-3)+(\kappa-2)(\kappa-3)\right] + (\gamma-2)(\kappa-2)(\gamma+\kappa-6) + \frac{1}{6}(\gamma-2)(\gamma-3)(\kappa-2)(\kappa-3)$.

Proof. Consider the gradient of $L_8(\mathbf{a},\mathbf{p}) = N_8(\mathbf{a},\mathbf{p}) + c(1-p_0-p_1-\cdots-p_{m_t})$.

$$
\begin{aligned}
&\nabla_{\mathbf{p}} L_8(\mathbf{a},\mathbf{p}) \\
=&\nabla_{\mathbf{p}}\left(N_8(\mathbf{a},\mathbf{p}) + c(1-p_0-p_1-\cdots-p_{m_t})\right) \\
=&w_1\left[\nabla_{\mathbf{p}}\left(f^2(X)f^2(X^{-1})\right)\right]_0 + w_2\left[\nabla_{\mathbf{p}}\left(f(X^2)f(X^{-2})f^2(X)f^2(X^{-1})\right)\right]_0 \\
&+ w_3\left[\nabla_{\mathbf{p}}\left(f(X^2)f^2(X)f^4(X^{-1})\right)\right]_0 + w_4\left[\nabla_{\mathbf{p}}\left(f^4(X)f^4(X^{-1})\right)\right]_0 - c\mathbf{1}_{m_t+1} \\
=&w_1\{\left[4f^2(X)f(X^{-1})(X^{-a_0},X^{-a_1},\ldots,X^{-a_{m_t}})\right])_0 \\
&+ \bar{w}_2\left[2f(X^2)f^2(X)f^2(X^{-1})(X^{-2a_0},\ldots,X^{-2a_{m_t}})\right]_0 \\
&+ \bar{w}_2\left[4f(X^2)f(X^{-2})f^2(X)f(X^{-1})(X^{-a_0},\ldots,X^{-a_{m_t}})\right]_0 \\
&+ \bar{w}_3\left[f^2(X)f^4(X^{-1})(X^{2a_0},X^{2a_1},\ldots,X^{2a_{m_t}})\right]_0 \\
&+ \bar{w}_3\left[2f(X^2)f(X)f^4(X^{-1})(X^{a_0},X^{a_1},\ldots,X^{a_{m_t}})\right]_0 \\
&+ \bar{w}_3\left[4f(X^2)f^2(X)f^3(X^{-1})(X^{-a_0},X^{-a_1},\ldots,X^{-a_{m_t}})\right]_0 \\
&+ \bar{w}_4\left[8f^4(X)f^3(X^{-1})(X^{-a_0},X^{-a_1},\ldots,X^{-a_{m_t}})\right]_0\} - c\mathbf{1}_{m_t+1}.
\end{aligned}
\tag{4.8}
$$

When $P_8(\mathbf{a},\mathbf{p})$ reaches its minimum, $\nabla_{\mathbf{p}}\left[L(\mathbf{a},\mathbf{p})\right] = \mathbf{0}_{m_t+1}$, which is equivalent to (7) by defining $c_0 = c/w_1$. □

Based on Lemma 6 and Lemma 7, we adopt the gradient-descent algorithm to obtain a locally optimal edge distribution for SC codes with coupling pattern $\mathbf{a}$, starting from the uniform distribution inside $\mathbf{P}$ as presented in Algorithm 4. Note that conv$(\cdot)$ and flip$(\cdot)$ refer to convolution and reverse of vectors, respectively.

4.4 Generalization of GRADE

We have explained the basic idea of GRADE in optimizing the edge distribution of SC code ensembles with respect to the expected number of cycles. Cycles have been studied

Algorithm 4 Gradient-Descent Distributor (GRADE) for Cycle Optimization

Inputs:

$\gamma, \kappa, m_t, m, \mathbf{a}$: parameters of the SC code;

w: weight of each cycle-6 candidate;

ϵ, α: accuracy and step size of gradient descent;

Outputs:

$\mathbf{p}$: a locally optimal edge distribution over vals($\mathbf{a}$);

1: $\bar{w}_1 \leftarrow \frac{2w}{3}(\gamma-2)(\kappa-2)$, obtain $\{\bar{w}_i\}_{i=2}^{4}$ in Lemma 7;
2: $v_{prev} = 1$; $v_{cur} = 1$;
3: $\mathbf{p}, \mathbf{g} \leftarrow \mathbf{0}_{m_t+1}$, $\mathbf{f}, \bar{\mathbf{f}} \leftarrow \mathbf{0}_{m+1}$, $\mathbf{f}_2, \bar{\mathbf{f}}_2 \leftarrow \mathbf{0}_{2m+1}$;
4: $\mathbf{p} \leftarrow \frac{1}{m_t+1}\mathbf{1}_{m_t+1}$;
5: $\mathbf{f}\left[a_0, \ldots, a_{m_t}\right] \leftarrow \mathbf{p}$, $\bar{\mathbf{f}} \leftarrow \text{flip}(\mathbf{f})$;
6: $\mathbf{f}_2\left[1, 3, \ldots, 2m+1\right] \leftarrow \mathbf{f}$, $\bar{\mathbf{f}}_2 \leftarrow \text{flip}(\mathbf{f}_2)$;
7: $\mathbf{q}_1 \leftarrow \bar{w}_1\text{conv}(\mathbf{f}, \mathbf{f}, \mathbf{f}, \bar{\mathbf{f}}, \bar{\mathbf{f}}, \bar{\mathbf{f}})$, $\mathbf{q}_2 \leftarrow \text{conv}(\mathbf{f}, \mathbf{f}, \bar{\mathbf{f}}, \bar{\mathbf{f}})$;
8: $\mathbf{q}_3 \leftarrow \bar{w}_2\text{conv}(\mathbf{f}_2, \bar{\mathbf{f}}_2, \mathbf{f}, \mathbf{f}, \bar{\mathbf{f}}, \bar{\mathbf{f}}) + \bar{w}_3\text{conv}(\mathbf{f}_2, \mathbf{f}, \mathbf{f}, \bar{\mathbf{f}}, \bar{\mathbf{f}}, \bar{\mathbf{f}}, \bar{\mathbf{f}}) + \bar{w}_4\text{conv}(\mathbf{f}, \mathbf{f}, \mathbf{f}, \mathbf{f}, \bar{\mathbf{f}}, \bar{\mathbf{f}}, \bar{\mathbf{f}}, \bar{\mathbf{f}})$;
9: $v_{prev} = v_{cur}$, $v_{cur} = \mathbf{q}_1\left[3m\right] + \mathbf{q}_2\left[2m\right] + \mathbf{q}_3\left[4m\right]$;
10: $\mathbf{g}_1 \leftarrow 6\bar{w}_1\text{conv}(\mathbf{f}, \mathbf{f}, \mathbf{f}, \bar{\mathbf{f}}, \bar{\mathbf{f}})$, $\mathbf{g}_2 \leftarrow 4\text{conv}(\mathbf{f}, \mathbf{f}, \bar{\mathbf{f}})$;
11: $\mathbf{g}_3 \leftarrow 4\bar{w}_2\text{conv}(\mathbf{f}_2, \bar{\mathbf{f}}_2, \mathbf{f}, \mathbf{f}, \bar{\mathbf{f}}) + 2\bar{w}_3\text{conv}(\bar{\mathbf{f}}_2, \mathbf{f}, \mathbf{f}, \mathbf{f}, \mathbf{f}, \bar{\mathbf{f}}) + 4\bar{w}_3\text{conv}(\mathbf{f}_2, \mathbf{f}, \mathbf{f}, \bar{\mathbf{f}}, \bar{\mathbf{f}}, \bar{\mathbf{f}}) + 8\bar{w}_4\text{conv}(\mathbf{f}, \mathbf{f}, \mathbf{f}, \mathbf{f}, \bar{\mathbf{f}}, \bar{\mathbf{f}}, \bar{\mathbf{f}})$;
12: $\mathbf{g}_4 \leftarrow 2\bar{w}_2\text{conv}(\mathbf{f}_2, \mathbf{f}, \mathbf{f}, \bar{\mathbf{f}}, \bar{\mathbf{f}}) + \bar{w}_3\text{conv}(\mathbf{f}, \mathbf{f}, \mathbf{f}, \mathbf{f}, \bar{\mathbf{f}}, \bar{\mathbf{f}})$;
13: $\mathbf{g} \leftarrow \mathbf{g}_1\left[2m + \mathbf{a}\right] + \mathbf{g}_2\left[m + \mathbf{a}\right] + \mathbf{g}_3\left[3m + \mathbf{a}\right] + \mathbf{g}_4\left[2m + 2\mathbf{a}\right]$, $\mathbf{g} \leftarrow \mathbf{g} - \text{mean}(\mathbf{g})$;
14: **if** $|v_{prev} - v_{cur}| > \epsilon$ **then**
15: $\quad \mathbf{p} \leftarrow \mathbf{p} - \alpha\frac{\mathbf{g}}{\|\mathbf{g}\|}$;
16: $\quad$ **goto** step 5;
17: **end if**
18: **return** $\mathbf{p}$;

extensively in related literature (see e.g., [76, 77, 78]) due to their simplicity and presence in problematic objects. However, cycles alone do not always account for typical decoding failures; for example, isolated cycles are not as harmful as concentrated cycles in codes with VN degree 4 since single cycles on their own do not lead to decoding failures (as captured by e.g., ASs [41]), rather concatenated cycles do. An excessive focus on the removal of isolated cycles can lead to remarkably less degrees of freedom for the removal of dominant problematic objects. In this section, we therefore extend the theory of GRADE to arbitrary subgraphs.

4.4.1 Probabilistic Metric

In this subsection, we generalize the results presented in Section 4.3.1 to obtain closed-form representations of the expected number of objects with arbitrary topologies. The key idea is that the dependency among nodes within each object can be fully described by a minimal set of fundamental cycles (or basic cycles), which is referred to as the **cycle basis** of the object (see Definition 12) [79, 80, 81].

Definition 12. *(**Cycle Basis**) A **cycle basis** of an object is a minimum-cardinality set of cycles using disjunctive unions of which, each cycle in the object can be obtained; we call the cycles in this set **fundamental cycles**.*

In the remainder of this chapter, we define a **prototype** of an object, for simplicity, as an assignment of the indices of its variable nodes (VNs) and its check nodes (CNs) in the base matrix. Prototype is a natural extension of *pattern*, i.e., a prototype of an object is exactly a pattern (discussed in [41]) when the object is a cycle. According to [81], we call a prototype **active** if all the fundamental cycles satisfy the cycle condition simultaneously, which means the detrimental object will be created in the protograph after partitioning the base matrix. The probability of a prototype becoming active under a random partition is proved to be represented by the constant term of a multi-variate polynomial, where each

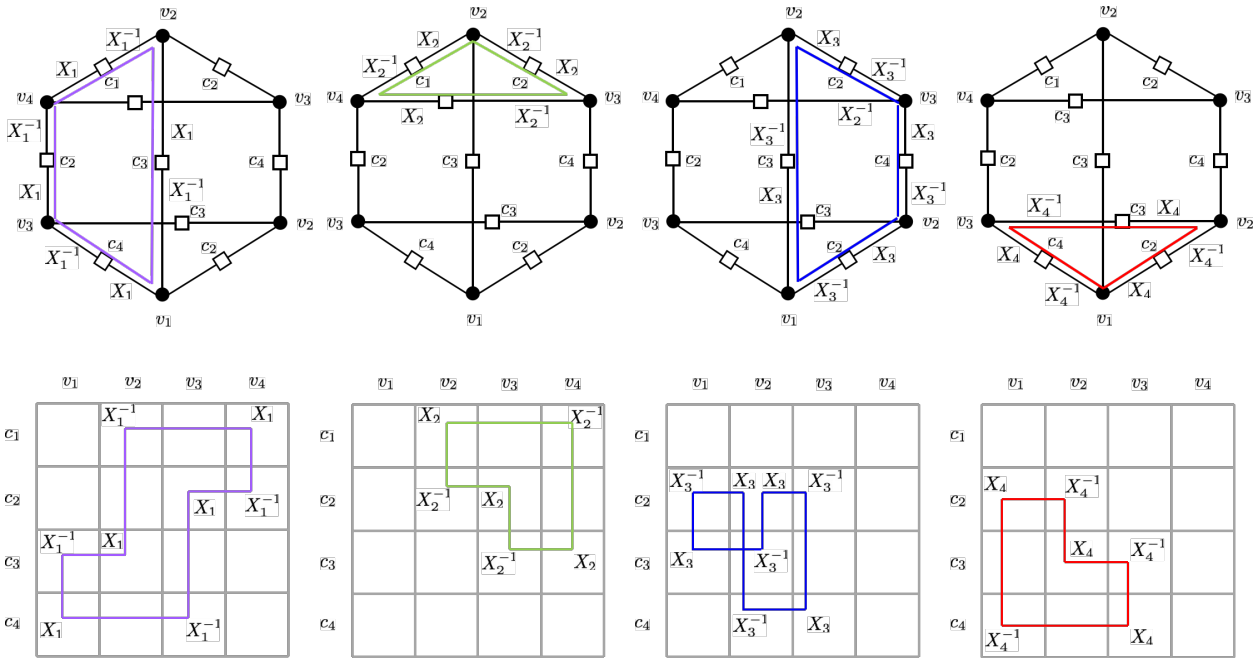


Figure 4.3: The cycle basis of a typical $(6, 0)$ $((6, 6))$-AS in SC codes with $\gamma = 3$ ($\gamma = 4$) and their corresponding cycle candidates while pulled back to the base matrix. The cycle basis has 4 fundamental cycles as shown in the top 4 panels; each cycle decides a cycle candidate in the base matrix and an independent variable in the characteristic polynomial, as shown in the bottom panels.

variable is associated with a cycle in the cycle basis: we refer to this polynomial as the **characteristic polynomial** of the object associated with fixed prototype. The overall characteristic polynomial of the object without specifying the prototype is then obtained as an average over the characteristic polynomials associated with all possible prototypes.

We start with a motivating example.

Example 18. *Take the object (AS) with the node assignment shown in the top panel of Fig. 4.3 as an example.*[1] *Consider the cycle basis consisting of the* 4 *cycles highlighted in Fig. 4.3 and their associated variables* X_i, $1 \leq i \leq 4$. *We refer to the cycle associated with* X_i, $1 \leq i \leq 4$, *as cycle* i. *In the bottom panel of Fig. 4.3, the labels* X_i *and* X_i^{-1} *are placed alternately on the cycle candidate corresponding to cycle* i *in the base matrix.*

[1]Note that we only keep nodes with intrinsic connections, i.e., we ignored the degree 1 CNs as they are not involved in any cycles and thus do not affect the probability of a prototype becoming active.

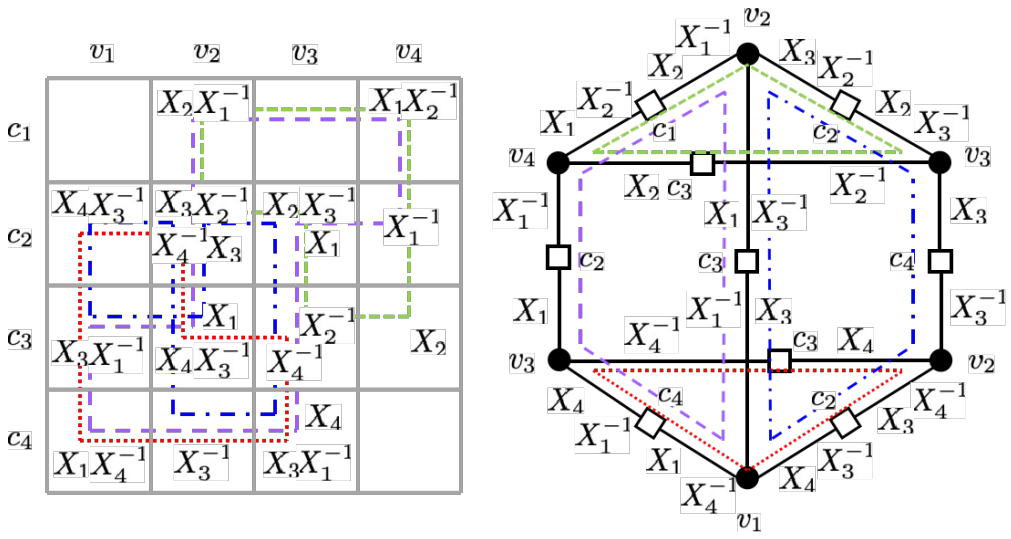


Figure 4.4: The matrix representation of the characteristic polynomial of the AS in Fig. 4.3. The monomial in each entry corresponds to a factor in the characteristic polynomial in (4.9).

We next briefly and intuitively explain how the characteristic polynomial of the object is specified as follows:

$$\begin{aligned} h(\mathbf{X}) =& f(X_2^{-1}X_3^2X_4^{-1})f(X_1X_2X_3^{-1})f(X_1X_3^{-1}X_4)f(X_1^{-1}X_3X_4) \\ & f(X_1X_2^{-1})f(X_1^{-1}X_2)f^2(X_1^{-1}X_3)f(X_1X_4^{-1})f(X_2^{-1}X_4^{-1})f(X_3^{-1}X_4) \\ & f(X_1^{-1})f(X_2)f(X_3^{-1}), \end{aligned} \tag{4.9}$$

where $f(\cdot)$ is the coupling polynomial of a cycle as specified in Definition 11.

As shown in Fig. 4.4, we place the labels on all the cycle candidates (see Fig. 4.3) altogether in the base matrix. Then, each entry of the matrix becomes associated with the product of all the labels contained in it. Take the entry at the intersection of row c_3 and column v_2 as an example. This entry is labeled with X_1, X_3^{-1}, X_4 on the cycle candidates for cycles 1, 3, and 4, respectively. Therefore, the entry is associated with $X_1X_3^{-1}X_4$. Each product is the monomial corresponding to the matrix entry. The characteristic polynomial in (4.9) is exactly the product of all the factors obtained by replacing the variable in the coupling polynomial by the monomials corresponding to each entry.

In a way similar to the process described in the proof of Theorem 6, expanding the right-

hand side (RHS) of (4.9) results in terms of the form $q_i X_1^{k_1} X_2^{k_2} X_3^{k_3} X_4^{k_4}$ for each, where q_i is the probability of a unique assignment to vertices, i.e., matrix entries, on the prototype of the AS such that the alternating sum of entries on cycle i associated with X_i in the cycle basis is k_i, $1 \leq i \leq 4$. Therefore, the constant term is exactly the sum of the probabilities of all possible assignments (of the partitioning matrix) such that the cycle candidates of all the fundamental cycles satisfy their cycle conditions (all k_i's are zeros). In other words, the constant term is exactly the probability of the prototype becoming active in the Tanner graph.

In Example 18, we have briefly introduced the idea of how we define the characteristic polynomial of an object associated with a fixed prototype. However, as shown in the case of cycle-8 candidates, an object is typically associated with multiple prototypes (referred to as cycle candidates when the object is a cycle). We next present an efficient method to obtain the expected number of all possible prototypes corresponding to an object.

The major idea is described as follows. Each prototype of an object leads to an equivalence relation on the CNs and VNs of the object, in which nodes with identical indices are regarded as being equivalent. The set consisting of all the prototypes describing the same equivalence relation is referred to as a **prototype class**. The characteristic polynomials of the prototypes belonging to the same prototype class are identical, and the cardinality of each prototype class is determined by their associated equivalence relation. Therefore, the key steps to obtain the characteristic polynomial of an object are: 1) to enumerate all the possible prototype classes of (or non-isomorphic equivalence relations on) a given object, and then 2) to obtain their associated characteristic polynomials and cardinalities.

For example, consider the prototype class described by the graph in the left panel of Fig. 4.5. Throughout this chapter, we use $[n]$ to represent the set $\{1, 2, \ldots, n\}$ for any $n \in \mathbb{N}$. Any assignment of $c_1, c_2, c_3, c_4 \in [\gamma]$ and $v_1, v_2, v_3, v_4 \in [\kappa]$ such that c_1, c_2, c_3, c_4 are mutually different, v_1, v_2, v_3, v_4 are also mutually different belongs to a unique prototype in the prototype class described by the graph. Note that the uniqueness follows from the fact that the **automorphism group** of the prototype class has only the identity element, thus

the cardinality of this prototype class is $4!\binom{\gamma}{4}4!\binom{\kappa}{4}$. The automorphism group of a prototype is defined later in Definition 14; however, the aforementioned cardinality intuitively implies that each row/column permutation results in a unique prototype. We then move on to obtain the characteristic polynomial directly through the prototype class. The equivalence relation on CNs and VNs induces an equivalence relation on edges, in which edges with VNs and CNs all from the same equivalence class are referred to as being equivalent. As shown in Fig. 4.5, edges from the same equivalence class are highlighted by identical markers.

Note that each equivalence class on edges corresponds to a unique entry in the base matrix. Recall that each entry is associated with the product of all labels contained in it, which corresponds to a separate factor in the characteristic polynomial. In a similar way, if we represent each equivalence class by the product of all labels on all edges contained in it, the resultant product will be exactly the monomial associated with the entry corresponding to this equivalence class. Therefore, each factor of the characteristic polynomial in (4.9) is associated with an equivalence class on edges. For example, in Fig. 4.5, the two edges highlighted by red triangles belong to the same equivalence class and are labeled with X_1 and $X_2X_3^{-1}$, respectively; and they altogether correspond to the factor $f(X_1X_2X_3^{-1})$ in the characteristic polynomial.

In the remaining text, we represent the equivalence relation by $\sim$.

Definition 13. *(**Prototype Class**) Let $\gamma, \kappa \in \mathbb{N}$. Consider an object represented by the bipartite graph $G(V, C, E)$, where V and C denote the set of VNs and CNs, respectively. The set E is the set of all edges, where each edge is represented by $e_{i,j}$, for some $i \in V$, $j \in C$, connecting nodes i and j. Let $\mathcal{V}, \mathcal{C}$ represent equivalence classes on V and C, respectively. A **prototype** is an assignment $P = (f, g)$, where $f : V \to [\kappa]$, $g : C \to [\gamma]$ such that:*

1. *For any $c \in C$ and $v_1, v_2 \in V$ such that $e_{v_1,c}, e_{v_2,c} \in E$, $f(v_1) \neq f(v_2)$;*

2. *For any $v \in V$ and $c_1, c_2 \in C$ such that $e_{v,c_1}, e_{v,c_2} \in E$, $g(c_1) \neq g(c_2)$.*

The set consisting of all the prototypes $P = (f, g)$ that satisfy the following conditions is

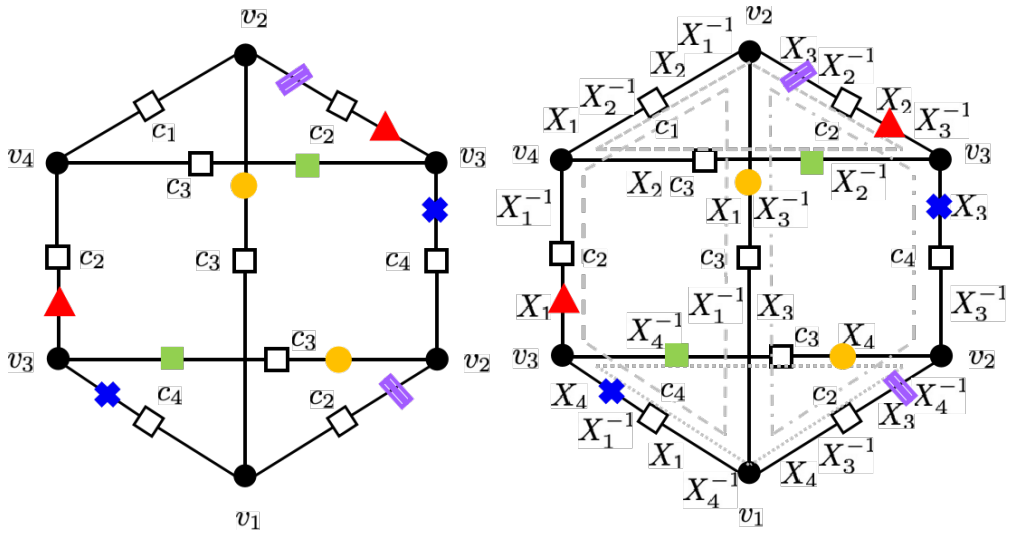


Figure 4.5: The graph representation of the characteristic polynomial of the AS in Fig. 4.3

referred to as the ***prototype class*** *associated with* $(\mathcal{V}, \mathcal{C})$*, denoted by* $\mathcal{P}(\mathcal{V}, \mathcal{C})$*:*

1. *For any* $v_1, v_2 \in V$*,* $f(v_1) = f(v_2)$ *iff.* $v_1 \sim v_2$ *in* $\mathcal{V}$ *(same column in the matrix);*

2. *For any* $c_1, c_2 \in C$*,* $g(c_1) = g(c_2)$ *iff.* $c_1 \sim c_2$ *in* $\mathcal{C}$ *(same row in the matrix).*

Denote the equivalence class induced by the relation: $e_{v_1,c_1} \sim e_{v_2,c_2}$ *iff.* $v_1 \sim v_2$ *and* $c_1 \sim c_2$*, for all* $v_1, v_2 \in V$ *and* $c_1, c_2 \in C$*, by* $\mathcal{E}(\mathcal{V}, \mathcal{C})$*, which is referred to as the equivalence class induced by* $\mathcal{V}$ *and* $\mathcal{C}$*.*

Lemma 8. *(**Characteristic Polynomial of Prototypes**) Consider the bipartite graph* $G(V, C, E)$ *of an object and the prototype class* $\mathcal{P}(\mathcal{V}, \mathcal{C})$*. Suppose* $\mathcal{E}(\mathcal{V}, \mathcal{C})$ *is the equivalence class on edges induced by* $\mathcal{V}$ *and* $\mathcal{C}$*.*

Let S *denote the cycle basis of* G*. Define* $\delta : E \times S \to \{-1, 0, 1\}$ *as follows: for any* $s \in S$*,* $s = (v_1, c_1, v_2, c_2, \ldots, v_g, c_g)$*, and* $e \in E$*,* $\delta_{e,s} = 1$ *if* $e = e_{v_i,c_i}$ *for some* $i \in [g]$*,* $\delta_{e,s} = -1$ *if* $e = e_{v_{i+1},c_i}$ *for some* $i \in [g]$*, otherwise* $\delta_{e,s} = 0$*.*

Define $h(\mathbf{X}; G|\mathcal{V}, \mathcal{C})$ *as a polynomial of* G *associated with* $\mathcal{P}(\mathcal{V}, \mathcal{C})$ *and given by:*

$$h(\mathbf{X}; G|\mathcal{V}, \mathcal{C}) = \prod_{\bar{e} \in \mathcal{E}(\mathcal{V}, \mathcal{C})} f\left(\prod_{e \in \bar{e}} \prod_{s \in S} X_s^{\delta_{e,s}} \right). \tag{4.10}$$

Then, the constant term of $h(\mathbf{X}; G|\mathcal{V}, \mathcal{C})$ is the probability that a prototype belonging to the class $\mathcal{P}(\mathcal{V}, \mathcal{C})$ is active in the Tanner graph after partitioning.

Proof. For simplicity, we write $\mathcal{E}$ instead of $\mathcal{E}(\mathcal{V}, \mathcal{C})$ in the proof. Any assignment on the set of edges E can be represented by $\mathbf{x} \in (x_1, x_2, \ldots, x_{|E|}) \in \text{vals}(\mathbf{a})^{|E|}$ ($\text{vals}(\mathbf{a})$ is defined in Theorem 6 as the set $\{a_0, a_1, \ldots, a_{m_t}\}$), where x_e denotes the assignment on edge e for any $e \in E$. Consider that all the edges belonging to the same equivalence class in $\mathcal{E}$ correspond to the same entry in the base matrix (and the partitioning matrix); these edges need to be assigned with an identical number in the partitioning matrix. Therefore, the assignment on the set of edges is essentially an assignment on the equivalence classes. Let $\mathbf{i} \in \{0, 1, \ldots, m_t\}^{|\mathcal{E}|}$ denote an assignment on the equivalence classes $\mathcal{E}$, where each element of $\mathbf{i}$ is represented by $i_{\bar{e}}$ for some $\bar{e} \in \mathcal{E}$; all the edges in the equivalence class $\bar{e}$ are assigned with $a_{i_{\bar{e}}}$ in the partitioning matrix, for any $\bar{e} \in \mathcal{E}$.

We know that

$$
\begin{aligned}
h(\mathbf{X}; G|\mathcal{V}, \mathcal{C}) &= \prod_{\bar{e} \in \mathcal{E}} f\left(\prod_{e \in \bar{e}} \prod_{s \in S} X_s^{\delta_{e,s}}\right) \\
&= \sum_{\mathbf{i} \in \{0,1,\ldots,m_t\}^{|\mathcal{E}|}} \prod_{\bar{e} \in \mathcal{E}} \left[p_{i_{\bar{e}}} \prod_{e \in \bar{e}} \prod_{s \in S} X_s^{\delta_{e,s} a_{i_{\bar{e}}}} \right] \\
&= \sum_{\mathbf{i} \in \{0,1,\ldots,m_t\}^{|\mathcal{E}|}} \left(\prod_{\bar{e} \in \mathcal{E}} p_{i_{\bar{e}}} \right) \prod_{\bar{e} \in \mathcal{E}} \prod_{e \in \bar{e}} \prod_{s \in S} X_s^{\delta_{e,s} a_{i_{\bar{e}}}} \\
&= \sum_{\mathbf{i} \in \{0,1,\ldots,m_t\}^{|\mathcal{E}|}} \left(\prod_{\bar{e} \in \mathcal{E}} p_{i_{\bar{e}}} \right) \prod_{s \in S} X_s^{\sum_{\bar{e} \in \mathcal{E}} a_{i_{\bar{e}}} \sum_{e \in \bar{e}} \delta_{e,s}} \\
&= \sum_{\mathbf{i} \in \{0,1,\ldots,m_t\}^{|\mathcal{E}|}} \left(\prod_{\bar{e} \in \mathcal{E}} p_{i_{\bar{e}}} \right) \prod_{s \in S} X_s^{l_s(\mathbf{i})},
\end{aligned}
\tag{4.11}
$$

where $l_s(\mathbf{i}) = \sum_{\bar{e} \in \mathcal{E}} a_{i_{\bar{e}}} \sum_{e \in \bar{e}} \delta_{e,s} = \sum_{e \in s, e \in \bar{e}} \delta_{e,s} a_{i_{\bar{e}}}$ is exactly the alternating sum of the assignment $\mathbf{i}$ on cycle s.

Denote by $\mathcal{Z}(G)$ the set of assignments of the prototype in the partitioning matrix such that this prototype becomes active in the Tanner graph of the code after partitioning. Then,

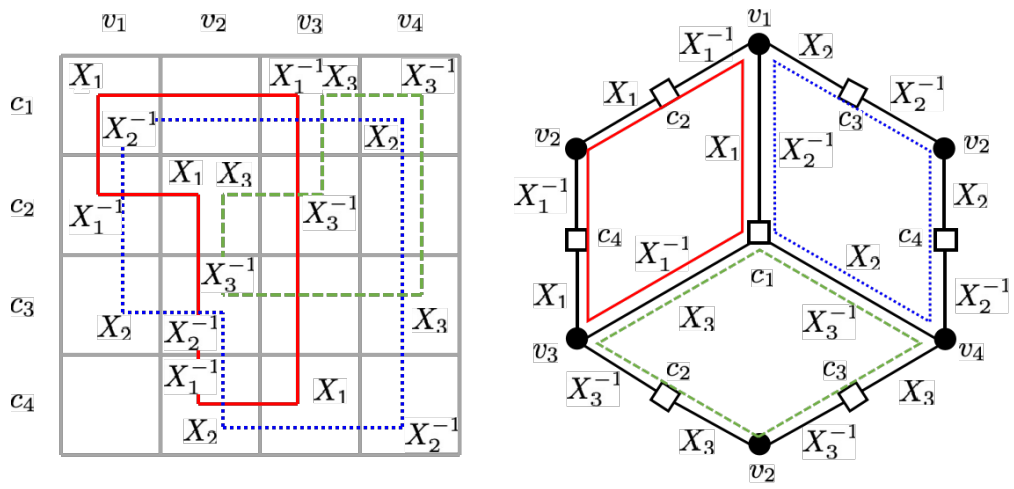


Figure 4.6: The matrix representation of the characteristic polynomial of the AS in Remark 7.

$$
\begin{aligned}
[h(\mathbf{X}; G|\mathcal{V}, \mathcal{C})]_0 = & \sum_{\mathbf{i}\in\{0,1,\ldots,m_t\}^{|\mathcal{E}|}: l_s(\mathbf{i})=0, \forall s\in S} \prod_{\bar{e}\in\mathcal{E}} p_{i_{\bar{e}}} \\
= & \sum_{\mathbf{i}\in\{0,1,\ldots,m_t\}^{|\mathcal{E}|}: l_s(\mathbf{i})=0, \forall s\in S} \mathbb{P}\left[x_e = a_{i_{\bar{e}}}, \forall \bar{e}\in\mathcal{E}, e\in\bar{e}\right] \qquad (4.12) \\
= & \mathbb{P}\left[\mathcal{Z}(G)\right],
\end{aligned}
$$

which indicates that the constant term of $h(\mathbf{X}; G|\mathcal{V}, \mathcal{C})$ is the probability we are seeking. □

In fact, the coefficients of other terms of $h(\mathbf{X}; G|\mathcal{V}, \mathcal{C})$ also specify the probabilities of partitioning assignments other than $\mathcal{Z}(G)$. Consequently, $h(\mathbf{X}; G|\mathcal{V}, \mathcal{C})$ in (4.10) represents the characteristic polynomial of G associated with $\mathcal{P}(\mathcal{V}, \mathcal{C})$.

While elementary objects (absorbing sets in particular) dominate the error floor of binary LDPC codes and NB-LDPC codes over the AWGN channel, non-elementary objects are observed to notably contribute to the error floor of NB-LDPC codes over non-canonical channels, e.g., practical magnetic recording and Flash channels [82, 72, 73].

Remark 7. *(**Non-Elementary Objects**) Note that Lemma 8 extends beyond elementary objects. Fig. 4.6 shows a prototype of a non-elementary object. This prototype can still be described by a set of* 3 *elementary cycles, as shown in Fig. 4.6 via colors. According to*

Lemma 8, the characteristic polynomial of the prototype is:

$$\begin{aligned} h(\mathbf{X}; G|\mathcal{V}, \mathcal{C}) =& f(X_1 X_2^{-1}) f(X_2 X_3^{-1}) f(X_3 X_1^{-1}) f(X_1 X_3) f(X_1^{-1} X_2) f(X_2^{-1} X_3^{-1}) \\ & f(X_1) f(X_1^{-1}) f(X_2) f(X_2^{-1}) f(X_3) f(X_3^{-1}). \end{aligned} \tag{4.13}$$

In combination with the WCM framework proposed in [73] that optimizes the edge weights of NB-LDPC codes on fixed unweighted graphs, our method can open a door to systematically optimizing NB-SC codes with high memories, which have potential to be adopted in storage systems among other applications.

After obtaining the characteristic polynomial of any object associated with a fixed prototype class, we proceed to obtain the expectation of the number of active prototypes over all prototype classes. The essential step here is to calculate the cardinality of each prototype class. A natural property here is that each prototype from a specific class corresponds to assigning non-repeated elements with order from $[\kappa]$ and $[\gamma]$ to the equivalence classes in $\mathcal{V}$ and $\mathcal{C}$, respectively. However, specific permutations of values assigned to the nodes can lead to some other assignments that are isomorphic to each other because of the intrinsic symmetry of the prototypes. For example, in Fig. 4.7(a), the assignment that exchanges values v_1 and v_2 while keeping values on remaining VNs as they are is equivalent to the original assignment. We call this exchange operation an **automorphism** over G under $\mathcal{P}(\mathcal{V}, \mathcal{C})$ and denote it by $(v_1 v_2)$. The automorphisms over G under each prototype class form a group, which is defined in Definition 14.

Definition 14. *(**Automorphism Group of an Object Under a Prototype Class**) For any object represented by a bipartite graph $G(V, C, E)$, let $\mathcal{P}(\mathcal{V}, \mathcal{C})$ be a prototype class of G. An **automorphism** over G under $\mathcal{P}(\mathcal{V}, \mathcal{C})$ is a pair of bijections (π_V, π_C) written as $\pi_V \pi_C$, where $\pi_V : V \to V$ and $\pi_C : C \to C$ are bijections such that*

1. *$\forall v \in V$, $c \in C$, $e_{v,c} \in E$ iff. $e_{\pi_V(v), \pi_C(c)} \in E$;*

2. *$\forall v_1, v_2 \in V$, $v_1 \sim v_2$ iff. $\pi_V(v_1) \sim \pi_V(v_2)$;*

3. $\forall c_1, c_2 \in C$, $c_1 \sim c_2$ *iff.* $\pi_C(c_1) \sim \pi_C(c_2)$.

The set containing all automorphisms over G *under* $\mathcal{P}(\mathcal{V}, \mathcal{C})$ *is referred to as the* ***automorphism group*** *of* G *under* $\mathcal{P}(\mathcal{V}, \mathcal{C})$.

Remark 8. *From Definition 14, we know that any automorphism over* G *under* $\mathcal{P}(\mathcal{V}, \mathcal{C})$ *preserves the equivalence relation specified by* $(\mathcal{V}, \mathcal{C})$, *i.e.,* $\{\pi_V(\bar{v}), \forall \bar{v} \in \mathcal{V}\} = \mathcal{V}$, $\{\pi_C(\bar{c}), \forall \bar{c} \in \mathcal{C}\} = \mathcal{C}$. *Therefore, each automorphism can be simply represented as a pair of permutations over* $\mathcal{V}$ *and* $\mathcal{C}$.

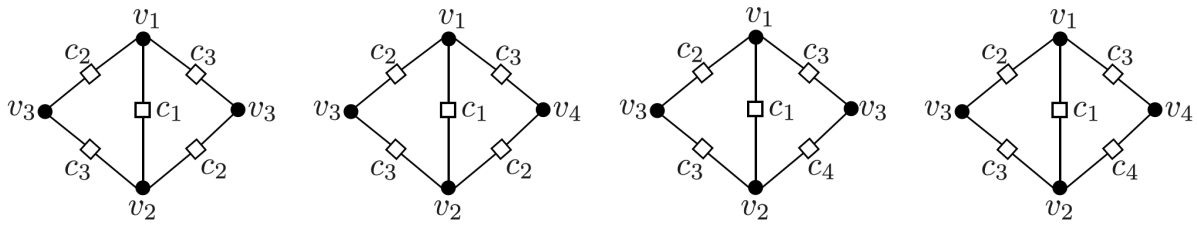
Lemma 9. *Given the bipartite graph* $G(V, C, E)$ *of an object, let* $\mathcal{B}(G)$ *denote the set consisting of all prototype classes of* G. *Define the characteristic polynomial* $h(\mathbf{X}; G)$ *of object* G *as follows:*

$$h(\mathbf{X}; G) = \sum_{\mathcal{P}(\mathcal{V},\mathcal{C}) \in \mathcal{B}(G)} \frac{|\mathcal{V}|!|\mathcal{C}|!}{|\mathrm{Aut}(G|\mathcal{V}, \mathcal{C})|} \binom{\kappa}{|\mathcal{V}|} \binom{\gamma}{|\mathcal{C}|} h(\mathbf{X}; G|\mathcal{V}, \mathcal{C}), \tag{4.14}$$

where $\mathrm{Aut}(G|\mathcal{V}, \mathcal{C})$ *denotes the automorphism group of the bipartite graph* G *under prototype class* $\mathcal{P}(\mathcal{V}, \mathcal{C})$. *Then,* $[h(\mathbf{X}; G)]_0$ *is exactly the expected number of active prototype of object* G.

Proof. There are $|\mathcal{V}|!|\mathcal{C}|!\binom{\kappa}{|\mathcal{V}|}\binom{\gamma}{|\mathcal{C}|}$ assignments on indices of nodes in G in the base matrix that satisfy the equivalence relation specified by $(\mathcal{V}, \mathcal{C})$. Among these assignments, each one has been counted exactly $|\mathrm{Aut}(G|\mathcal{V}, \mathcal{C})|$ times. Therefore, the cardinality of the prototype class $\mathcal{P}(\mathcal{V}, \mathcal{C})$ is exactly $\frac{|\mathcal{V}|!|\mathcal{C}|!}{|\mathrm{Aut}(G|\mathcal{V},\mathcal{C})|}\binom{\kappa}{|\mathcal{V}|}\binom{\gamma}{|\mathcal{C}|}$.

For any prototype P of G, define a Bernoulli random variable X_P, where $\mathbb{P}[X_P = 1] = \mathbb{P}[P \text{ is active}]$, $\mathbb{P}[X_P = 0] = \mathbb{P}[P \text{ is not active}]$. Let $X = \sum_P X_P$ denotes the summation of X_P over all possible prototypes of G. Then,



(a) Prototype class 1. (b) Prototype class 2. (c) Prototype class 3. (d) Prototype class 4.

Figure 4.7: The 4 prototype classes of 2 concatenated cycles-6 and their corresponding topologies.

$$
\begin{aligned}
\mathbb{E}[X] &= \sum_{P} \mathbb{E}[X_P] \\
&= \sum_{\mathcal{P}(\mathcal{V},\mathcal{C})\in\mathcal{B}(G)} \sum_{P\in\mathcal{P}(\mathcal{V},\mathcal{C})} \mathbb{E}[X_P] \\
&= \sum_{\mathcal{P}(\mathcal{V},\mathcal{C})\in\mathcal{B}(G)} \sum_{P\in\mathcal{P}(\mathcal{V},\mathcal{C})} \mathbb{P}[X_P = 1] \\
&= \sum_{\mathcal{P}(\mathcal{V},\mathcal{C})\in\mathcal{B}(G)} \sum_{P\in\mathcal{P}(\mathcal{V},\mathcal{C})} [h(\mathbf{X}; G|\mathcal{V},\mathcal{C})]_0 \\
&= \sum_{\mathcal{P}(\mathcal{V},\mathcal{C})\in\mathcal{B}(G)} \frac{|\mathcal{V}|!|\mathcal{C}|!}{|\mathrm{Aut}(G|\mathcal{V},\mathcal{C})|} \binom{\kappa}{|\mathcal{V}|} \binom{\gamma}{|\mathcal{C}|} [h(\mathbf{X}; G|\mathcal{V},\mathcal{C})]_0 \\
&= [h(\mathbf{X}; G)]_0 .
\end{aligned}
\tag{4.15}
$$

Thus, the lemma is proved. □

Example 19. *Suppose $\gamma \in \{3, 4\}$. Take the graph G of two concatenated cycles-6 as an example; there are 4 different possible prototype classes $(\mathcal{V}_i, \mathcal{C}_i)$, $1 \le i \le 4$, as shown in Fig. 4.7. The automorphism groups corresponding to the 4 prototype classes, denote by*

$|\mathrm{Aut}(G|\mathcal{V}_i,\mathcal{C}_i)|$, $1 \le i \le 4$, *respectively, are:*

$$\begin{aligned}
&\mathrm{Aut}(G|\mathcal{V}_1,\mathcal{C}_1) = \{e,(v_1v_2),(c_2c_3),(v_1v_2)(c_1c_3)\},\\
&\mathrm{Aut}(G|\mathcal{V}_2,\mathcal{C}_2) = \{e,(v_1v_2)(c_2c_3),(v_3v_4)(c_2c_3),(v_1v_2)(v_3v_4)\},\\
&\mathrm{Aut}(G|\mathcal{V}_3,\mathcal{C}_3) = \{e,(v_1v_2)(c_2c_4)\},\\
&\mathrm{Aut}(G|\mathcal{V}_4,\mathcal{C}_4) = \{e,(v_1v_2)(v_3v_4)(c_2c_4)\},
\end{aligned} \tag{4.16}$$

where the element e in these groups is the identity element (no permutations). Therefore, the cardinality of the automorphism groups corresponding to the 4 prototype classes are $|\mathrm{Aut}(G|\mathcal{V}_1,\mathcal{C}_1)| = 4$, $|\mathrm{Aut}(G|\mathcal{V}_2,\mathcal{C}_2)| = 4$, $|\mathrm{Aut}(G|\mathcal{V}_3,\mathcal{C}_3)| = 2$, *and* $|\mathrm{Aut}(G|\mathcal{V}_4,\mathcal{C}_4)| = 2$, *respectively. Moreover, the characteristic polynomials for G corresponding to each prototype class are:*

$$\begin{aligned}
h(\mathbf{X};G|\mathcal{V}_1,\mathcal{C}_1) =& f(X_1X_2)f(X_1^{-1}X_2^{-1})f(X_1X_2^{-1})f(X_1^{-1}X_2)f(X_1)f(X_1^{-1})f(X_2)f(X_2^{-1}),\\
h(\mathbf{X};G|\mathcal{V}_3,\mathcal{C}_3) =& f(X_1X_2)f(X_1^{-1}X_2^{-1})f(X_1^{-1}X_2)f^2(X_1)f(X_1^{-1})f(X_2)f^2(X_2^{-1}),\\
h(\mathbf{X};G|\mathcal{V}_2,\mathcal{C}_2) =& h(\mathbf{X};G|\mathcal{V}_4,\mathcal{C}_4) = f(X_1X_2)f(X_1^{-1}X_2^{-1})f^2(X_1)f^2(X_1^{-1})f^2(X_2)f^2(X_2^{-1}).
\end{aligned} \tag{4.17}$$

According to Lemma 9, when $\gamma = 3$, *the characteristic polynomial* $h(\mathbf{X};G)$ *is derived as follows:*

$$\begin{aligned}
&h(\mathbf{X};G)\\
=&\frac{\kappa!\gamma!}{4(\kappa-3)!(\gamma-3)!}f(X_1X_2)f(X_1^{-1}X_2^{-1})f(X_1X_2^{-1})f(X_1^{-1}X_2)f(X_1)f(X_1^{-1})f(X_2)f(X_2^{-1})\\
&+\frac{\kappa!\gamma!}{4(\kappa-4)!(\gamma-3)!}f(X_1X_2)f(X_1^{-1}X_2^{-1})f^2(X_1)f^2(X_1^{-1})f^2(X_2)f^2(X_2^{-1})\\
=&\frac{3\kappa!}{2(\kappa-4)!}\Bigg(f(X_1X_2)f(X_1^{-1}X_2^{-1})f^2(X_1)f^2(X_1^{-1})f^2(X_2)f^2(X_2^{-1})\\
&+\frac{1}{\kappa-3}f(X_1X_2)f(X_1^{-1}X_2^{-1})f(X_1X_2^{-1})f(X_1^{-1}X_2)f(X_1)f(X_1^{-1})f(X_2)f(X_2^{-1})\Bigg).
\end{aligned} \tag{4.18}$$

When $\gamma = 4$, the characteristic polynomial $h(\mathbf{X}; G)$ is derived as follows:

$$
\begin{aligned}
& h(\mathbf{X}; G) \\
=& \frac{\kappa!\gamma!}{4(\kappa-3)!(\gamma-3)!} f(X_1X_2)f(X_1^{-1}X_2^{-1})f(X_1X_2^{-1})f(X_1^{-1}X_2)f(X_1)f(X_1^{-1})f(X_2)f(X_2^{-1}) \\
& + \frac{\kappa!\gamma!}{2(\kappa-3)!(\gamma-4)!} f(X_1X_2)f(X_1^{-1}X_2^{-1})f(X_1^{-1}X_2)f^2(X_1)f(X_1^{-1})f(X_2)f^2(X_2^{-1}) \\
& + \frac{\kappa!\gamma!}{4(\kappa-4)!(\gamma-3)!} f(X_1X_2)f(X_1^{-1}X_2^{-1})f^2(X_1)f^2(X_1^{-1})f^2(X_2)f^2(X_2^{-1}) \\
& + \frac{\kappa!\gamma!}{2(\kappa-4)!(\gamma-4)!} f(X_1X_2)f(X_1^{-1}X_2^{-1})f^2(X_1)f^2(X_1^{-1})f^2(X_2)f^2(X_2^{-1}) \\
=& \frac{18\kappa!}{(\kappa-4)!} \Bigg(f(X_1X_2)f(X_1^{-1}X_2^{-1})f^2(X_1)f^2(X_1^{-1})f^2(X_2)f^2(X_2^{-1}) \\
& + \frac{2}{3(\kappa-3)} f(X_1X_2)f(X_1^{-1}X_2^{-1})f(X_1^{-1}X_2)f^2(X_1)f(X_1^{-1})f(X_2)f^2(X_2^{-1}) \\
& + \frac{1}{3(\kappa-3)} f(X_1X_2)f(X_1^{-1}X_2^{-1})f(X_1X_2^{-1})f(X_1^{-1}X_2)f(X_1)f(X_1^{-1})f(X_2)f(X_2^{-1}) \Bigg).
\end{aligned}
\tag{4.19}
$$

Remark 9. *Observe that in Example 19, the number of assignments such that all edges are distinct dominates among all the cases, especially when κ is large enough; we refer to such dominant assignments as the* ***typical assignments****. Therefore, it is normally sufficiently accurate to optimize over the characteristic polynomial corresponding to the typical assignments only. Specifically, for 2 concatenated cycles of length $2i$ and $2j$, suppose the number of edges in common is $2k$. Then, the characteristic polynomial can be well approximated by $\tilde{h}(\mathbf{X}; G) = Cf^k(X_1X_2)f^k(X_1^{-1}X_2^{-1})f^{i-k}(X_1)f^{i-k}(X_1^{-1})f^{j-k}(X_2)f^{j-k}(X_2^{-1})$ for some constant $C \in \mathbb{N}$.*

4.4.2 Gradient-Descent Distributor

Theorem 8. *Given the bipartite graph $G(V, C, E)$ of an object and the prototype class $\mathcal{P}(\mathcal{V}, \mathcal{C})$. Suppose $\mathcal{E}(\mathcal{V}, \mathcal{C})$ is the equivalence class induced by $\mathcal{V}$ and $\mathcal{C}$. Let $(\mathbf{v})_t$ be the t-th entry of the vector $\mathbf{v}$. Following the notation in Lemma 8, the gradient of $[h(\mathbf{X}; G|\mathcal{V}, \mathcal{C})]_0$*

with respect to $\mathbf{p}$ *is given by:*

$$\left(\nabla_{\mathbf{p}}\left[h(\mathbf{X};G|\mathcal{V},\mathcal{C})\right]_0\right)_t = \left[\sum_{\bar{e}\in\mathcal{E}}\prod_{s\in S} X_s^{a_t\sum_{e\in\bar{e}}\delta_{e,s}} \prod_{\bar{e}'\in\mathcal{E}\setminus\{\bar{e}\}} f\left(\prod_{e\in\bar{e}'}\prod_{s\in S} X_s^{\delta_{e,s}}\right)\right]_0. \tag{4.20}$$

Proof. We obtain the gradient with respect to $\mathbf{p}$ as follows:

$$\begin{aligned}
&\left(\nabla_{\mathbf{p}} h(\mathbf{X};G|\mathcal{V},\mathcal{C})\right)_t \\
=&\sum_{\bar{e}\in\mathcal{E}}\left(\nabla_{\mathbf{p}} f\left(\prod_{e\in\bar{e}}\prod_{s\in S} X_s^{\delta_{e,s}}\right)\right)_t \prod_{\bar{e}'\in\mathcal{E}\setminus\{\bar{e}\}} f\left(\prod_{e\in\bar{e}'}\prod_{s\in S} X_s^{\delta_{e,s}}\right) \\
=&\sum_{\bar{e}\in\mathcal{E}}\left(\nabla_{\mathbf{p}} f\left(\prod_{s\in S}\prod_{e\in\bar{e}} X_s^{\delta_{e,s}}\right)\right)_t \prod_{\bar{e}'\in\mathcal{E}\setminus\{\bar{e}\}} f\left(\prod_{e\in\bar{e}'}\prod_{s\in S} X_s^{\delta_{e,s}}\right) \\
=&\sum_{\bar{e}\in\mathcal{E}}\left(\nabla_{\mathbf{p}} f\left(\prod_{s\in S} X_s^{\sum_{e\in\bar{e}}\delta_{e,s}}\right)\right)_t \prod_{\bar{e}'\in\mathcal{E}\setminus\{\bar{e}\}} f\left(\prod_{e\in\bar{e}'}\prod_{s\in S} X_s^{\delta_{e,s}}\right) \\
=&\sum_{\bar{e}\in\mathcal{E}}\prod_{s\in S} X_s^{a_t\sum_{e\in\bar{e}}\delta_{e,s}} \prod_{\bar{e}'\in\mathcal{E}\setminus\{\bar{e}\}} f\left(\prod_{e\in\bar{e}'}\prod_{s\in S} X_s^{\delta_{e,s}}\right).
\end{aligned} \tag{4.21}$$

Therefore,

$$\begin{aligned}
&\left(\nabla_{\mathbf{p}}\left[h(\mathbf{X};G|\mathcal{V},\mathcal{C})\right]_0\right)_t = \left[\left(\nabla_{\mathbf{p}} h(\mathbf{X};G|\mathcal{V},\mathcal{C})\right)_t\right]_0 \\
=&\left[\sum_{\bar{e}\in\mathcal{E}}\prod_{s\in S} X_s^{a_t\sum_{e\in\bar{e}}\delta_{e,s}} \prod_{\bar{e}'\in\mathcal{E}\setminus\{\bar{e}\}} f\left(\prod_{e\in\bar{e}'}\prod_{s\in S} X_s^{\delta_{e,s}}\right)\right]_0.
\end{aligned} \tag{4.22}$$

□

Provided the explicit expression of $h(\mathbf{X};G)$ and its gradient, one can easily apply the gradient-descent algorithm to obtain an edge distribution that locally minimizes the expected number of active prototypes of the object specified by G. In Example 20 and Example 21, we apply GRADE to two concatenated cycles-8 as shown in Fig. 4.8, under any prototype class $\mathcal{P}(\mathcal{V},\mathcal{C})$ such that $\mathcal{V}$ and $\mathcal{C}$ induce no equivalent edges in E. Denote by $P_{8-8}(\mathbf{a},\mathbf{p}|\mathcal{V},\mathcal{C})$ the probability that a prototype of this object becomes active after partitioning in an SC

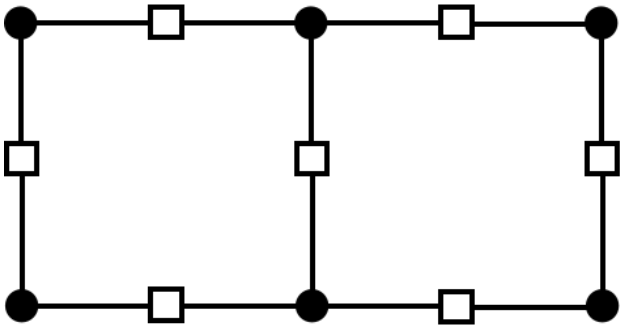


Figure 4.8: The targeted object consisting of two concatenated cycle-8 in Example 20 and Example 21.

ensemble with coupling pattern $\mathbf{a}$ and edge distribution $\mathbf{p}$.

Example 20. *Consider the following three cases of SC ensembles with* $m = 6$*:*

1. *Full-memory codes with uniform edge distribution:* $m_t = m = 6$*,* $\mathbf{a} = (0, 1, \ldots, 6)$ *and* $\mathbf{p} = \frac{1}{7}\mathbf{1}_7$*. Then,* $P_{8-8}(\mathbf{a}, \mathbf{p}|\mathcal{V}, \mathcal{C}) = 0.0049$*;*

2. *Full-memory codes with distribution obtained from GRADE:* $m_t = m = 6$*,* $\mathbf{a} = (0, 1, \ldots, 6)$ *and* $\mathbf{p} = (0.2991, 0.0899, 0.0749, 0.0733, 0.0749, 0.0896, 0.2984)$*. Then,* $P_{8-8}(\mathbf{a}, \mathbf{p}|\mathcal{V}, \mathcal{C}) = 0.0032$*;*

3. *Non-full-memory codes with distribution obtained from GRADE:* $m_t = 3$*,* $\mathbf{a} = (0, 1, 4, 6)$ *and* $\mathbf{p} = (0.2604, 0.2063, 0.2219, 0.3114)$*. Then,* $P_{8-8}(\mathbf{a}, \mathbf{p}|\mathcal{V}, \mathcal{C}) = 0.0035$*.*

Example 21. *Consider the following three cases of SC ensembles with* $m = 9$*:*

1. *Full-memory codes with uniform edge distribution:* $m_t = m = 9$*,* $\mathbf{a} = (0, 1, \ldots, 9)$ *and* $\mathbf{p} = \frac{1}{10}\mathbf{1}_{10}$*. Then,* $P_{8-8}(\mathbf{a}, \mathbf{p}|\mathcal{V}, \mathcal{C}) = 0.0024$*;*

2. *Full-memory codes with distribution obtained from GRADE:* $m_t = m = 9$*,* $\mathbf{a} = (0, 1, \ldots, 9)$ *and* $\mathbf{p} = (0.2648, 0.0803, 0.0509, 0.0526, 0.0519, 0.0519, 0.0525, 0.0508, 0.0801, 0.2644)$*. Then,* $P_{8-8}(\mathbf{a}, \mathbf{p}|\mathcal{V}, \mathcal{C}) = 0.0015$*;*

3. *Non-full-memory codes with distribution obtained from GRADE:* $m_t = 4$*,* $\mathbf{a} = (0, 1, 4, 7, 9)$ *and* $\mathbf{p} = (0.2479, 0.1799, 0.1262, 0.1645, 0.2814)$*. Then,* $P_{8-8}(\mathbf{a}, \mathbf{p}|\mathcal{V}, \mathcal{C}) = 0.0016$*.*

Remark 10. *In contrast to what we have shown regarding applying GRADE to single cycles, the gains obtained from applying GRADE to concatenated cycles are much more evident. As shown in Example 20 and Example 21, for* $m = 6$ *and* $m = 9$*, the local minima obtained for full memory codes are quite close to the gains obtained for codes with coupling patterns* $(0, 1, 4, 6)$ *and* $(0, 1, 4, 7, 9)$*, respectively (referred to as* ***topologically-coupled (TC) codes*** *later on). We show next in Section 4.5 and Section 4.6 that TC codes have close performance to GD codes with full-memories, where both are obtained from applying GRADE-AO followed by CPO to concatenated cycles.*

Remark 11. *Note that although we focus on non-tail-biting SC codes throughout this chapter, this condition is by no means necessary. To extend our method to tail-biting codes, one just needs to change the cycle condition in (5.9) from "*$\sum_{k=1}^{g} \mathbf{P}(i_k, j_k) = \sum_{k=1}^{g} \mathbf{P}(i_k, j_{k+1})$*" to "*$\sum_{k=1}^{g} \mathbf{P}(i_k, j_k) \equiv \sum_{k=1}^{g} \mathbf{P}(i_k, j_{k+1}) \mod L$*". If* $L > m + 1$*, which is the typical case, the resultant optimal distribution is still very likely to be nonuniform because of the asymmetry among components indexed by* $\{0, 1, \ldots, m\}$*. This fact is important since while constructing non-tail-biting codes with large* κ *and* m*, a large* L *is typically desired due to the notable rate loss resulting from a small* L*. However, in certain practical applications, codes are typically of moderate length, which implies that a moderate* L *is desirable. In such situations where* κ *and* m *are large, tail-biting codes do not suffer the same rate loss, and they can offer high error floor performance despite limiting the gain obtained from threshold saturation.*

4.5 Algorithmic Optimization

We have developed the theory and the algorithm to obtain edge distributions that locally minimize the number of short cycles in Section 4.3.2 and generalized the results from cycles to arbitrary objects in Section 4.4.2. In this section, we investigate algorithmic optimizers (AO) that search for excellent partitioning matrices under the guidance of GRADE. In particular, the edge distribution $\mathbf{p}_{opt}$ obtained through GRADE confines the search space to

only contain matrices that have edge distributions near $\mathbf{p}_{opt}$.

We discuss both heuristic AOs based on semi-greedy algorithms and globally-optimal AOs based on variations of the OO technique proposed in [40, 41]. The heuristic AOs require low computational complexity and are applicable to arbitrary objects and any code parameters, but are only locally optimal. The OO-based AOs obtain the globally-optimal solutions, but currently only work on short cycles in SC codes with small pseudo-memories; we refer to these codes as **topologically-coupled (TC) codes**. The reason behind the nomenclature "TC codes" is the topological degrees of freedom they offer the code designer via the selection of the non-zero component matrices.

4.5.1 Heuristic AO

In this subsection, we consider AOs that are based on heuristic methods. In this case, our proposed GRADE algorithm obtains an edge distribution to guide the AO. Starting from a random partitioning matrix $\mathbf{P}$ with the derived distribution, one can perform a semi-greedy algorithm that searches for partitioning matrix near the initial $\mathbf{P}$ that locally minimizes the number of targeted objects. Constraining the search space to contain $\mathbf{P}$'s that have distributions within small L_1 and L_∞ distances from that of the original $\mathbf{P}$, and adopting the CPO next, significantly reduces the computational complexity to find a strong high-memory code. GRADE-guided heuristic AO has advantages in two aspects: 1) low complexity by reduced search space, and 2) higher probability of arriving at superior solution by providing a good enough initialization to AO that can avoid undesirable local minima.

Cycle-Based Optimization

Based on the GRADE specified in Algorithm 4, we first present in Algorithm 5 a corresponding AO that focuses on minimizing the weighted sum of the number of cycles-6 and cycles-8. We refer to codes obtained from GRADE-AO as **gradient-descent (GD) codes**. By replacing the initial distribution $\mathbf{p}$ with the uniform distribution, we obtain the so-called

Algorithm 5 Cycle-Based GRADE-A Optimizer (AO)

Inputs:

$\gamma, \kappa, m, m_t, \mathbf{a}$: parameters of an SC ensemble;
$\mathbf{p}$: edge distribution obtained from Algorithm 4;
w: weight of each cycle-6 assuming that of a cycle-8 is 1;
d_1, d_2: parameters indicating the size of the search space;

Outputs:

$\mathbf{P}$: a locally optimal partitioning matrix;

Obtain the lists $\mathcal{L}_6(i,j)$, $\mathcal{L}_8(i,j)$ of cycles-6 candidates and cycle-8 candidates in the base matrix that contain node (i,j), $1 \leq i \leq \gamma$, $1 \leq j \leq \kappa$;
Obtain $\mathbf{u} = \arg\min_{\mathbf{x} \in \{0,1,\ldots,\gamma\kappa\}^{m+1}, ||\mathbf{x}||_1 = \gamma\kappa} ||\frac{1}{\gamma\kappa}\mathbf{x} - \mathbf{p}||_2$;
for $i \in \{0, 1, \ldots, m\}$ **do**
    Place $\mathbf{u}[i+1]$ i's into $\mathbf{P}$ randomly;
end for
$\mathbf{d} \leftarrow \mathbf{0}_{m+1}$;
noptimal $\leftarrow$ False;
for $i \in \{1, 2, \ldots, \gamma\}$, $j \in \{1, 2, \ldots, \kappa\}$ **do**
    $n_6 \leftarrow |\mathcal{L}_6(i,j)|$, $n_8 \leftarrow |\mathcal{L}_8(i,j)|$, $n \leftarrow wn_6 + n_8$;
    for $v \in \{0, 1, \ldots, m\}$ **do**
        $\mathbf{d}' = \mathbf{d}$, $\mathbf{d}'[v+1] \leftarrow \mathbf{d}'[v+1] + 1$, $p \leftarrow \mathbf{P}(i,j)$;
        if $||\mathbf{d}'||_1 \leq d_1$ and $||\mathbf{d}'||_\infty \leq d_2$ **then**
            $\mathbf{P}(i,j) \leftarrow v$;
            $t_6 \leftarrow |\mathcal{L}_6(i,j)|$, $t_8 \leftarrow |\mathcal{L}_8(i,j)|$, $t \leftarrow wt_6 + t_8$;
            if $t < n$ **then**
                noptimal $\leftarrow$ True, $n \leftarrow t$, $\mathbf{d} \leftarrow \mathbf{d}'$, $\mathbf{P}(i,j) \leftarrow v$;
            end if
        end if
    end for
end for
if noptimal **then**
    goto step 6;
end if
return $\mathbf{P}$;

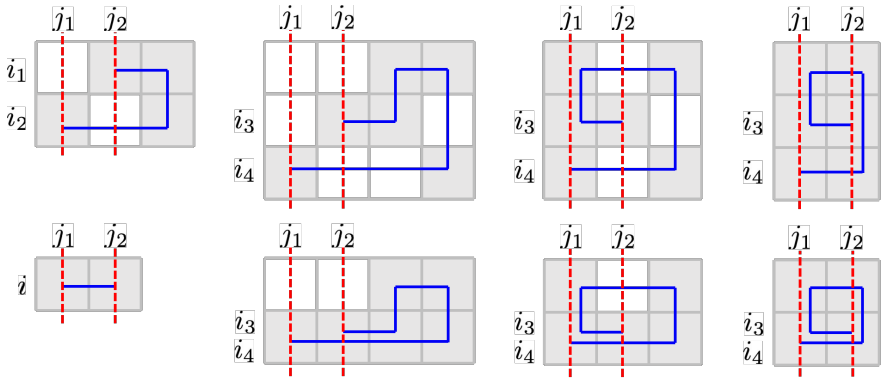


Figure 4.9: Type 1–3 paths mentioned in Algorithm 6: $\mathcal{L}_1(i, j_1, j_2)$, $\mathcal{L}_2(i_1, i_2, j_1, j_2)$, and $\mathcal{L}_3(i_3, i_4, j_1, j_2)$, $1 \leq j_1 < j_2 \leq \kappa$, $1 \leq i, i_1, i_2, i_3, i_4 \leq \gamma$, $i_1 \neq i_2$.

uniform (UNF) codes. In the special cases where the SC codes are not of full memory, i.e., the pseudo-memory is not identical with the memory, we refer to the codes obtained from GRADE-AO as **topologically-coupled (TC) codes**. We show in Section 4.6 by simulation that the distribution obtained by GRADE results in constructions that are better than those adopting uniform distribution and in existing literature.

Finer-Grained Optimization

We next develop a finer-grained optimizer in Algorithm 6, in which the targeted objects are two concatenated cycles where each of them is a cycle-6 or a cycle-8, as discussed in the examples of Section 4.4.2. The critical part of the algorithm is enumerating all the objects of interest efficiently. Since we focus on *concatenated* cycles of length 6 or 8, the key idea is to characterize concatenated cycles by the positions of the two degree 3 VNs and the three paths connecting them. Here, we only consider objects that are elementary ASs. We call a path $\mathcal{P} = v_1\text{-}c_1\text{-}v_2\text{-}\cdots\text{-}c_l\text{-}v_{l+1}$ a type-l path connecting (c_1, v_1) and (c_l, v_{l+1}) and denote it by $L(\mathcal{P}) = l$. Paths of type-1, type-2, and type-3 are shown in Fig. 4.9. Each concatenation of two cycles can be referred to as an i-j-k object, where i, j, k are the types of the three paths connecting the degree 3 nodes in this object. Our targeted objects, where each of the two cycles is either a cycle-6 or a cycle-8, can only be 2-1-2, 2-1-3, 2-2-2, or 3-1-3 objects. Steps 1-5 in Algorithm 6 are aimed at listing the paths of type 1–3, and all the possible

combinations of indices of the beginning and the ending CNs on each path.

Remark 12. *Note that in the finer-grained optimization, the condition of a concatenated-cycle pair in the protograph becoming a pair of concatenated cycles in the Tanner graph after lifting is that the two cycle candidates contained in this prototype all satisfy the cycle condition on lifting parameters specified in Lemma 4. Therefore, after applying AO to minimize the number of concatenated-cycle pairs, instead of using the original CPO designed for cycle optimization in [81], we adopt a modified version that is tailored for optimization over the number of pairs of concatenated cycles accordingly.*

In a way similar to what we have done with approaches eliminating cycles, we define GD codes, UNF codes, and TC codes here. Moreover, as shown in Section 4.4.2, the expected number of concatenated-cycle pairs in GD codes (with full memories) is close to that of GD-TC codes with carefully chosen pseudo-memories and coupling patterns.[2] In particular, memory 6 GD ensembles can be approximated by GD-TC ensembles with pseudo-memory 3 and coupling pattern $(0, 1, 4, 6)$; memory 9 GD ensembles can be approximated by GD-TC ensembles with pseudo-memory 4 and coupling pattern $(0, 1, 4, 7, 9)$. This leads to the conjecture that the performance of GD-TC codes and the performance of GD codes are quite close, which is somewhat surprising provided that they differ a lot in edge distribution, and the impact of concatenated cycles on waterfall performance is not strictly characterized. In Section 4.6.2, Monte-Carlo simulations support our conjecture, which enables more possibilities in TC codes. For example, TC codes can be globally-optimized given that their pseudo-memories are low; details will be discussed later on in Section 4.5.2.

4.5.2 Globally-Optimal AO

In this subsection, we explore globally-optimal constructions of TC codes with small pseudo-memories. The motivation behind this task is to construct an SC code with memory m

[2]We refer to the prototype of a pair of concatenated cycles as a concatenated-cycle pair.

Algorithm 6 Fine-Grained GRADE-A Optimizer (AO)

Inputs:

$\gamma, \kappa, m, m_t, \mathbf{a}$: parameters of an SC code with full memory;

$\mathbf{p}$: edge distribution obtained from Algorithm 4;

$\mathbf{w} = (w_1, w_2, w_3, w_4)$: the weights of 2-1-2, 2-1-3, 2-2-2, 3-1-3 objects, respectively.

d_1, d_2: parameters indicating the size of the search space;

Outputs:

$\mathbf{P}$: a locally optimal partitioning matrix;

Obtain the lists $\mathcal{L}_1(i, j_1, j_2)$, $\mathcal{L}_2(i_1, i_2, j_1, j_2)$, and $\mathcal{L}_3(i_3, i_4, j_1, j_2)$, $1 \leq j_1 < j_2 \leq \kappa$, $1 \leq i, i_1, i_2, i_3, i_4 \leq \gamma$, $i_1 \neq i_2$, where the lists are specified as follows:

a) $\mathcal{L}_1(i, j_1, j_2)$: all type-1 paths connecting (i, j_1) and (i, j_2) in the base matrix;

b) $\mathcal{L}_2(i_1, i_2, j_1, j_2)$: all type-2 paths connecting (i_1, j_1) and (i_2, j_2) in the base matrix;

c) $\mathcal{L}_3(i_3, i_4, j_1, j_2)$: all type-3 paths connecting (i_3, j_1) and (i_4, j_2) in the base matrix;

$\mathcal{I}_{212} \leftarrow \{(i, i_1, i_2, i_3, i_4) : 1 \leq i, i_1, i_2, i_3, i_4 \leq \gamma, i_1 < i_3, i_1 \neq i_2, i_3 \neq i_4\}$;

$\mathcal{I}_{213} \leftarrow \{(i, i_1, i_2, i_3, i_4) : 1 \leq i, i_1, i_2, i_3, i_4 \leq \gamma, i_1 \neq i_2\}$;

$\mathcal{I}_{222} \leftarrow \{(i_1, i_2, i_3, i_4, i_5, i_6) : 1 \leq i_1, i_2, i_3, i_4, i_5, i_6 \leq \gamma, i_1 < i_3 < i_5, i_1 \neq i_2, i_3 \neq i_4, i_5 \neq i_6\}$;

$\mathcal{I}_{313} \leftarrow \{(i, i_1, i_2, i_3, i_4) : 1 \leq i, i_1, i_2, i_3, i_4 \leq \gamma, i_1 < i_3\}$;

Obtain $\mathbf{u} = \arg\min_{\mathbf{x} \in \{0,1,2,\ldots,\gamma\kappa\}^{m+1}, ||\mathbf{x}||_1 = \gamma\kappa} ||\frac{1}{\gamma\kappa}\mathbf{x} - \mathbf{p}||_2$;

for $i \in \{0, 1, \ldots, m\}$ **do**

    Place $\mathbf{u}[i+1]$ i's into $\mathbf{P}$ randomly;

end for

$\mathbf{d} \leftarrow \mathbf{0}_{m+1}$;

$n \leftarrow M$; //M is some very large constant

noptimal $\leftarrow$ False;

for $i \in \{1, 2, \ldots, \gamma\}$, $j \in \{1, 2, \ldots, \kappa\}$ **do**

    for $v \in \{0, 1, \ldots, m\}$ **do**

        $\mathbf{d}' = \mathbf{d}$, $\mathbf{d}'[v+1] \leftarrow \mathbf{d}'[v+1] + 1$, $p \leftarrow \mathbf{P}(i, j)$;

        if $||\mathbf{d}'||_1 \leq d_1$ and $||\mathbf{d}'||_\infty \leq d_2$ **then**

            $\mathbf{P}(i, j) \leftarrow v$;

            for $1 \leq j_1 < j_2 \leq \kappa$, $1 \leq i_0, i_1, i_2, i_3, i_4 \leq \gamma$, $i_1 \neq i_2$ **do**

                $v_{1,l}(i_0, j_1, j_2) \leftarrow |\{\mathcal{P} | L(\mathcal{P}; \mathbf{P}) = l, \mathcal{P} \in \mathcal{L}_1(i_0, j_1, j_2)\}|$, $-m \leq l \leq m$;

                $v_{2,l}(i_1, i_2, j_1, j_2) \leftarrow |\{\mathcal{P} | L(\mathcal{P}; \mathbf{P}) = l, \mathcal{P} \in \mathcal{L}_2(i_1, i_2, j_1, j_2)\}|$, $-2m \leq l \leq 2m$;

                $v_{3,l}(i_3, i_4, j_1, j_2) \leftarrow |\{\mathcal{P} | L(\mathcal{P}; \mathbf{P}) = l, \mathcal{P} \in \mathcal{L}_3(i_3, i_4, j_1, j_2)\}|$, $-m \leq l \leq m$;

            end for

            $t_{212} \leftarrow \sum_{1 \leq j_1 < j_2 \leq \kappa} \sum_{(i_0, i_1, i_2, i_3, i_4) \in \mathcal{I}_{212}} \sum_{-m \leq l \leq m} v_{1,l}(i_0, j_1, j_2) v_{2,l}(i_1, i_2, j_1, j_2) v_{2,l}(i_3, i_4, j_1, j_2)$;

            $t_{213} \leftarrow \sum_{1 \leq j_1 < j_2 \leq \kappa} \sum_{(i_0, i_1, i_2, i_3, i_4) \in \mathcal{I}_{213}} \sum_{-m \leq l \leq m} v_{1,l}(i_0, j_1, j_2) v_{2,l}(i_1, i_2, j_1, j_2) v_{3,l}(i_3, i_4, j_1, j_2)$;

25: $t_{222} \leftarrow \sum_{1\le j_1<j_2\le\kappa} \sum_{(i_1,i_2,i_3,i_4,i_5,i_6)\in\mathcal{I}_{222}} \sum_{-2m\le l\le 2m} v_{2,l}(i_1,i_2,j_1,j_2)v_{2,l}(i_3,i_4,j_1,j_2)v_{2,l}(i_5,i_6,j_1,j_2)$;

26: $t_{313} \leftarrow \sum_{1\le j_1<j_2\le\kappa} \sum_{(i_0,i_1,i_2,i_3,i_4)\in\mathcal{I}_{313}} \sum_{-m\le l\le m} v_{1,l}(i_0,j_1,j_2)v_{3,l}(i_1,i_2,j_1,j_2)v_{3,l}(i_3,i_4,j_1,j_2)$;

27: $t \leftarrow w_1 t_{212} + w_2 t_{213} + w_3 t_{222} + w_4 t_{313}$;

28: **if** $t < n$ **then**

29: noptimal $\leftarrow$ True, $n \leftarrow t$, $\mathbf{d} \leftarrow \mathbf{d}'$, $\mathbf{P}(i,j) \leftarrow v$;

30: **end if**

31: **end if**

32: **end for**

33: **end for**

34: **if** noptimal **then**

35: **goto** step 11;

36: **end if**

37: **return** $\mathbf{P}$;

under the same computational complexity needed to construct a full memory m_t code, where $m_t < m$. Given m_t and m, we first find the optimal $\mathbf{a}$, in terms of the minimum number of prototypes of interest, with length $m_t + 1$ in a brute-force manner. Taking $m = 4$ and $m_t = 2$ as an example, the optimal coupling pattern with respect to the number of cycles is $\mathbf{a} = (0, 1, 4)$ and the corresponding optimal distribution is almost uniform. Moreover, we already know from Section 4.4.2 that regarding the optimal coupling pattern with respect to the number of concatenated-cycle pairs, $\mathbf{a} = (0, 1, 4, 6)$ and $\mathbf{a} = (0, 1, 4, 7, 9)$ are not only the optimal coupling patterns for $(m, m_t) = (6, 3)$ and $(m, m_t) = (9, 4)$, respectively, but also approximate the optimal full memory GD ensembles quite closely in terms of performance.

Given the optimal coupling pattern $\mathbf{a}$, we then obtain an optimal partitioning matrix by the OO method proposed in [40] and [41]. We extend the OO method for memory m_0 SC codes to any TC code with pseudo-memory $m_t = m_0$, which does not increase the complexity of the approach. Note that despite the current OO works only on cycles, future steps can be taken towards the extension of OO into concatenated cycles, which has potential to lead to TC codes with excellent performance that is even better than the GD codes with full memories, provided that it is much harder to obtain globally-optimal solutions for GD codes with full memories.

Optimal TC codes with pseudo-memory m_t have strictly fewer cycle candidates in their protographs than optimal SC codes with full memory $m = m_t$. Take $m = 4$ and $m_t = 2$ as an example. Suppose the optimal SC code has the partition $\mathbf{\Pi} = \mathbf{H}_0^{\mathrm{P}} + \mathbf{H}_1^{\mathrm{P}} + \mathbf{H}_2^{\mathrm{P}}$. Consider the TC code with partition $\mathbf{\Pi} = \mathbf{H}_0^{\mathrm{P}} + \mathbf{H}_1^{\mathrm{P}} + \mathbf{H}_4^{\mathrm{P}}$ such that $\mathbf{H}_2^{\mathrm{P}} = \mathbf{H}_4^{\mathrm{P}}$. Then, any cycle-6 candidate resulting from a cycle candidate in the base matrix assigned with 0-1-0-1-2-0, 1-2-1-2-2-0, or 0-1-2-1-x-x, $x \in \{0, 1, 2\}$, in $\mathbf{P}$ no longer has a counterpart in the TC code, since by replacing 2's with 4's, assignments 0-1-0-1-4-0, 1-4-1-4-4-0, and 0-1-4-1-x-x, $x \in \{0, 1, 4\}$, no longer satisfy the cycle condition in Lemma 4. Moreover, there exists a bijection between the remaining candidates in the SC code and all candidates in the TC code through the replacement of 2's with 4's.

Fig. 4.10 presents part of the protograph of a TC code with coupling pattern $(0, 1, 4)$ and that of its corresponding SC code with full memory 2. The cycle-6 candidate colored by blue is assigned with 0-1-2-1-1-1 in the SC code, which satisfies the cycle condition cycle candidates are generated in the protograph. However, the assignment becomes 0-1-4-1-1-1 in the TC code, which no longer satisfies the cycle condition and no cycle candidates are generated in the protograph. The cycle-6 candidate colored by green corresponds to one that results in cycle-6 candidates in both the SC and the TC codes shown in the figure. We also marked out a cycle-8 candidate (colored by red) that only leads to cycle candidates in the SC code.

According to the aforementioned discussion, TC codes are better (have less cycles) than SC codes with the same circulant size and $m = m_t$. In Section 4.6, we present simulation results of such codes and show that they can also outperform SC codes with the same constraint length (larger circulant size) and $m = m_t$.

4.6 Simulation Results

In this section, we show the frame error rate/uncorrectable bit error rate (FER/UBER) curves of seven groups of SC codes designed by the GRADE-AO methods presented in Sec-

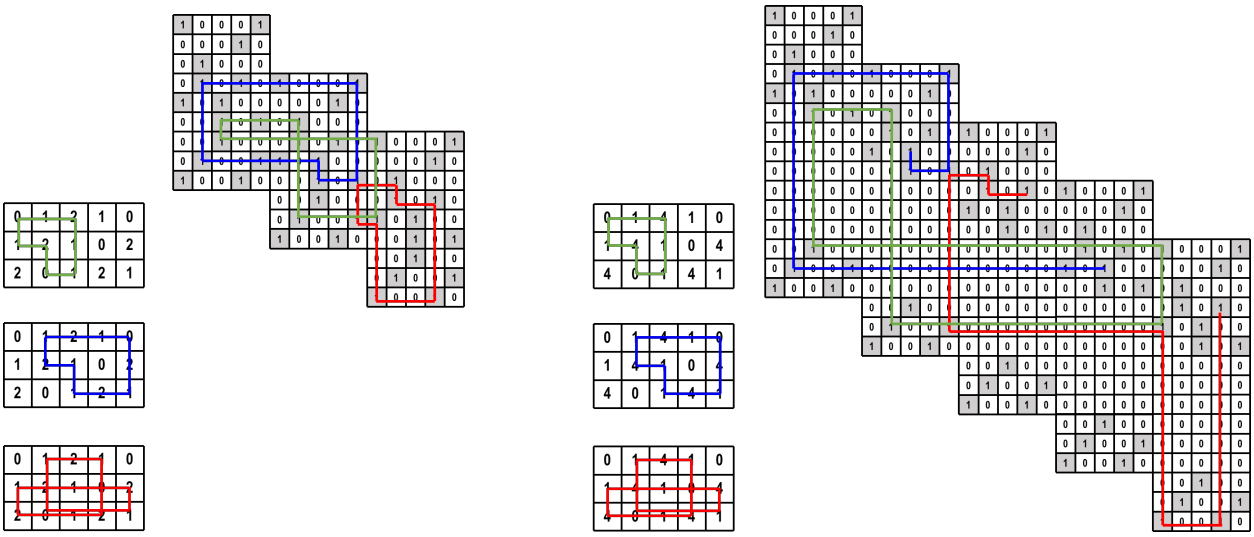


Figure 4.10: The first 5 replicas of the protograph of a TC code with coupling pattern $(0, 1, 4)$ (the right panel), and the first 3 replicas of the protograph of its corresponding SC code with memomry 2 (the left panel). Three cycle candidates (colored by green, blue, and red, respectively) in the base matrix and their corresponding paths in the two protographs are marked out.

tion 4.5, with respect to raw bit error rate/signal-to-noise ratio (RBER/SNR). We demonstrate that codes constructed by the GRADE-AO methods offer significant performance gains compared with codes with uniform edge distributions and codes constructed through purely algorithmic methods.

4.6.1 Optimization over Cycles

In this subsection, we simulate codes constructed based on optimizing the number of cycles using GRADE-AO specified in Section 4.3 on the AWGN channel. Out of these three plots, Fig. 4.11 and Fig. 4.12 compare GD codes with UNF codes designed as in Section 4.5.1. Fig. 4.13 compares a TC code designed as in Section 4.5.2 with optimal SC codes constructed through the OO-CPO method proposed in [40]. The GD/UNF codes have parameters $(\gamma, \kappa, m, z, L) = (3, 7, 5, 13, 100)$, $(3, 17, 9, 7, 100)$, and $(4, 29, 19, 29, 20)$, respectively. The TC code has parameters $(\gamma, \kappa, m_t, z, L) = (4, 17, 2, 17, 50)$ with the coupling pattern $\mathbf{a} = (0, 1, 4)$. For a fair comparison, we have selected two SC codes: one with a similar

Table 4.1: Statistics of the Number of Cycles

(γ, κ)	Code	Cycles-6	Cycles-8
$(3, 7)$	GD	0	0
	UNF	0	6,292
$(3, 17)$	GD	0	397,880
	UNF	0	559,902
	Battaglioni *et al.*[45]	0	451,337
$(4, 29)$	GD	0	528,090
	UNF	0	1,087,268
$(4, 17)$	TC	15,436	-
	SC (matched constraint length)	19,180	-
	SC (matched circulant size)	74,579	-

constraint length $(m + 1)z$ and the other with an identical circulant power z. To ensure that the SC codes and the TC code have close rates and codelengths, the two SC codes have parameters $(\gamma, \kappa, m, z, L) = (4, 17, 2, 28, 30)$ and $(4, 17, 2, 17, 50)$, respectively. The statistics regarding the number of cycles of each code are presented in Table 4.1.

Fig. 4.11 shows FER curves of our GD/UNF comparisons with $(\gamma, \kappa) = (3, 7)$ and $(3, 17)$. The partitioning matrices and the lifting matrices of the codes are specified in Section 4.6.3. When $\gamma = 3$, cycles-6 are easily removed by the CPO. Therefore, we perform joint optimization on the number of cycles-6 and cycles-8 candidates by assigning different weights to cycle candidates in Algorithm 5. We observe a performance gain for the GD code with respect to the UNF code in both the waterfall region and the error floor region. Moreover, the number of cycles-8 in the $(3, 17)$ GD code is reduced by 29% and 12% compared with the UNF code and the code constructed by Battaglioni *et al.* in [45], respectively. In addition, the $(3, 17)$ GD code has no weight-6 absorbing sets (ASs) and 133 weight-7 ASs, whereas the UNF code has 6 weight-6 ASs and 361 weight-7 ASs. As for the $(3, 7)$ codes, all cycles-6 and cycles-8 are removed. Thus, the gain of the GD code compared with the UNF code exceeds the gain observed in the $(3, 17)$ codes.

Fig. 4.12 shows FER curves of the GD/UNF comparison with $(\gamma, \kappa) = (4, 29)$. The partitioning matrices and the lifting matrices of the codes are specified in Section 4.6.3.

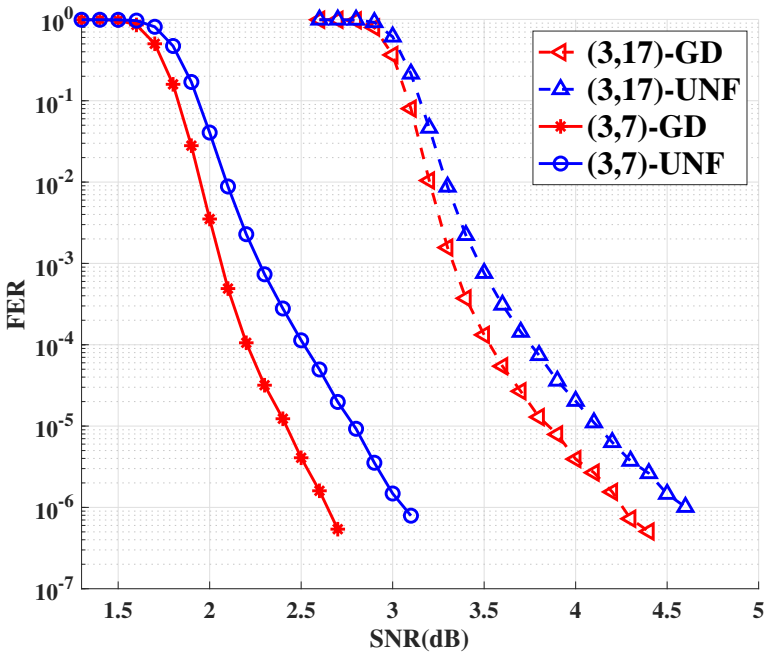


Figure 4.11: FER curves of GD/UNF codes with $\gamma = 3$ in the AWGN channel.

Cycles-6 in the GD code and the UNF code are both removed, and the number of cycles-8 in the GD code demonstrates a 51.4% reduction from the count observed in the UNF code. It is worth mentioning that both codes have no ASs of weights up to 8, which is reflected in their FER curves via the sharp waterfall regions and the non-existing error floor regions. The FER of the GD/UNF codes decreases with a rate exceeding 12 orders of magnitude per 0.5 dB signal-to-noise ratio (SNR) increase. Moreover, the GD code has a significant gain of about 0.25 dB over the UNF code.

Fig. 4.13 shows the FER curves of the TC/SC codes with $(\gamma, \kappa) = (4, 17)$. The partitioning matrices and the lifting matrices of the codes are specified in Section 4.6.3. The number of cycles-6 in the $(4, 17)$ TC code demonstrates a 79% and a 20% reduction from the counts observed in the SC codes with a matched constraint length and a matched circulant size, respectively. Moreover, the TC code has no weight-6 nor weight-8 ASs. It is shown that the TC code outperforms the optimal SC code with a matched constraint length, and that the gain is of greater magnitude when compared with the SC code of identical circulant size.

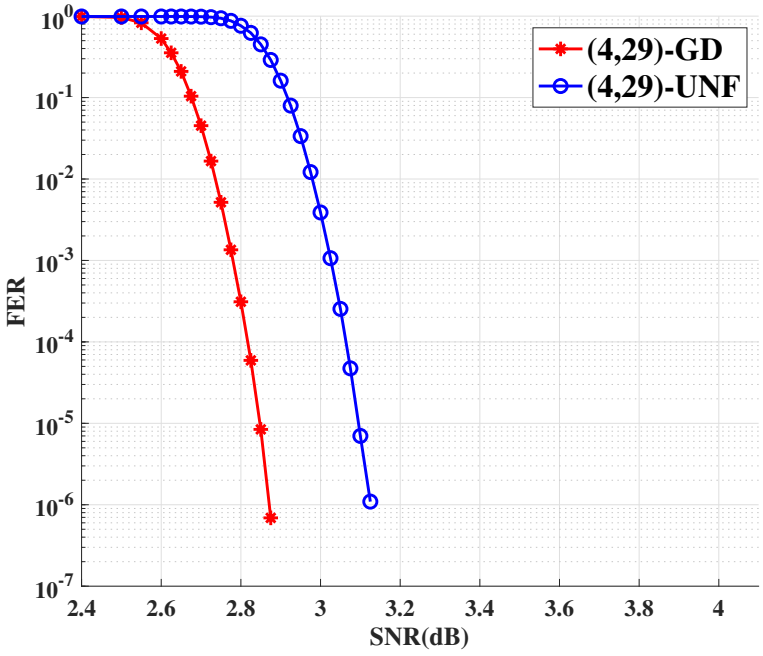


Figure 4.12: FER curves of GD/UNF codes with $(\gamma, \kappa) = (4, 29)$ in the AWGN channel.

Remark 13. *Note that although TC codes have higher memories and thus larger constraint lengths than SC codes of matched circulant sizes, they possess the same number of nonzero component matrices, and thus the same degrees of freedom in construction. This fact makes TC codes even more promising if we can devise for them windowed decoding algorithms with window sizes that are comparable to the corresponding SC codes of matched circulant sizes.*

4.6.2 Optimization over Concatenated Cycles

In this subsection, we simulate codes constructed based on minimizing the number of concatenated-cycle pairs using the GRADE-AO specified in Section 4.4 and Section 4.5.1. We compare GD/UNF codes with $m = 6$ in addition to TC codes with $m_t = 3$ and $\mathbf{a} = (0, 1, 4, 6)$ on the binary symmetric channel (BSC), Flash channel, and magnetic recording channel. There are two groups of GD/TC/UNF codes that have parameters $(\gamma, \kappa, m, z, L) = (4, 24, 6, 17, 40)$ and $(4, 20, 6, 13, 20)$, respectively. The statistics regarding the number of cycles of each code are presented in Table 4.2.

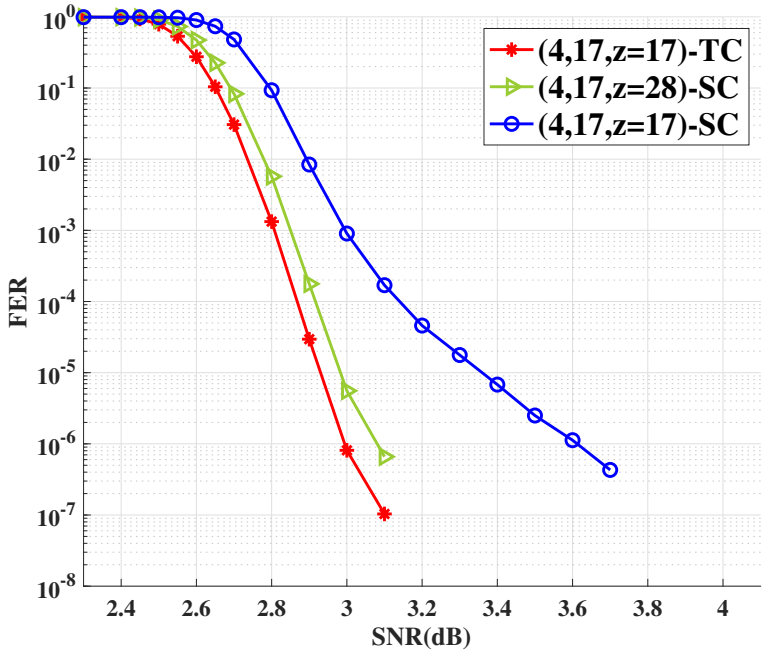


Figure 4.13: FER curves of TC/SC codes with $(\gamma, \kappa) = (4, 17)$ in the AWGN channel.

Table 4.2: Statistics of the Number of Concatenated Cycles

(γ, κ)	Code	Cycles-6	2-1-2 Objects	2-2-2 Objects	2-1-3 Objects	3-1-3 Objects
$(4, 24)$ (NLM)	GD	4,794	1,751	807,534	1,162,035	125,869,717
	TC	4,352	4,301	816,816	1,125,400	126,903,436
	UNF	11,713	7,293	1,308,762	2,615,297	208,425,933
$(4, 24)$ (BSC)	GD	4,794	1,802	822,120	1,212,100	126,514,918
	TC	4,403	4,097	807,534	1,139,000	126,434,525
	UNF	11,713	7,293	1,308,762	2,615,297	208,425,933
$(4, 20)$	GD	2,171	338	178,334	238,160	19,638,372
	TC	2,665	1,404	178,828	262,509	20,571,499
	UNF	2,444	2,860	287,014	540,865	34,362,393

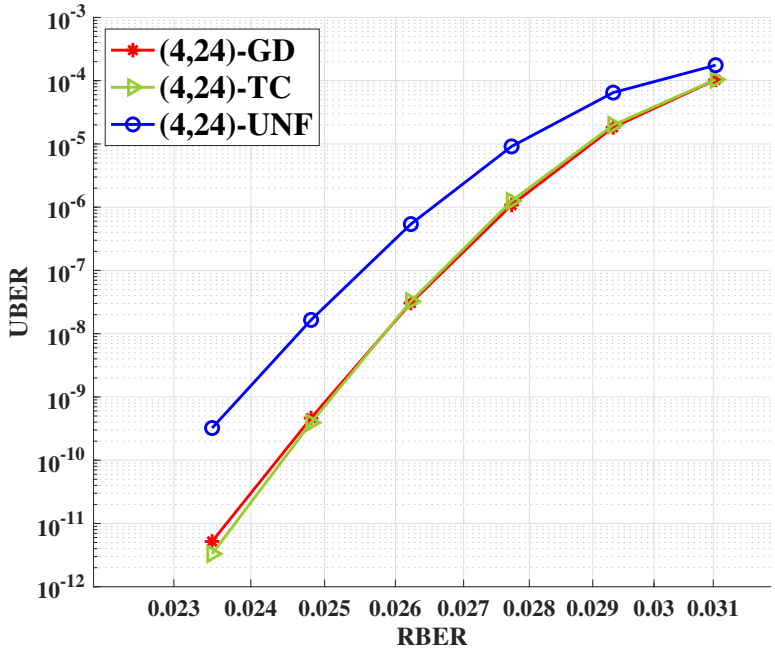


Figure 4.14: UBER curves of GD/TC/UNF codes with $(\gamma, \kappa) = (4, 24)$ in the NLM channel.

Fig. 4.14 shows UBER curves of the GD/TC/UNF codes with $(\gamma, \kappa) = (4, 24)$ on a Flash channel.[3] The Flash channel used in this section is a practical, asymmetric Flash channel, which is the normal-Laplace mixture (NLM) Flash channel [84]. 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/erasing problems, thereby resulting in significant asymmetry. Furthermore, the authors of [84] provided accurate fitting results of their model for program/erase (P/E) cycles up to 10 times the manufacturer's endurance specification (up to 30,000 P/E cycles). We implemented the NLM channel based on the parameters described in [84]. Here, we use 3 threshold voltage reads, and the sector size is 512 bytes. For decoding, we use a finite-precision (FP) fast Fourier transform based q-ary sum-product algorithm (FFT-QSPA) LDPC decoder [85]. The decoder performs a

[3]Note that although we only provide simulation results on some typical channels for brevity in this chapter, our approach is generally applicable to many other channels, such as the ones underlying three-dimensional cross point (3D XPoint) [36] and two-dimensional magnetic recording (TDMR) [83] systems.

maximum of 50 iterations, and it stops if a codeword is reached sooner.

The partitioning matrices and the lifting matrices of the codes are specified in Section 4.6.3. The non-binary edge weights are set as in [86] and [87]. The codes can be further optimized by applying the more advanced WCM framework presented in [72] and [73]. The first row of Table 4.2 shows the statistics of unlabeled cycles and concatenated cycles in each code. The number of objects in GD/TC codes are reduced by around 40% compared with the UNF code. No error floors are observed in any one of the UBER curves. The UBER of the GD/TC codes decreases with a rate exceeding 14 orders of magnitude per 0.01 RBER decrease. Moreover, the GD/TC codes have a significant gain of about 2 orders of magnitude over the UNF code at RBER 0.0235. It worths mentioning that the UBER curves of the GD/TC codes nearly overlap, which is in accordance with the closeness of the statistics of objects in them.

While NB codes are adopted in the simulations over the NLM channel, independent coding are more widely applied in practical Flash solutions in order to preserve high access speed. Therefore, we next present in Fig. 4.15 the UBER curves of the GD/TC/UNF codes with $(\gamma, \kappa) = (4, 24)$ on the BSC, as a simplified model of single-level cell (SLC) channel with 1 threshold voltage read.

While partitioning matrices of the $(4, 24)$ NB codes constructed for the NLM channels are adopted as they are here, we have modified the lifting parameters slightly in order to remove all unlabeled ASs with weights less than or equal to 7 in the GD/TC codes: this is achieved by changing one entry in each lifting matrix. The new lifting matrices of the GD/TC codes are specified in Section 4.6.3. According to Table 4.2, the number of objects in the GD/TC codes has not changed dramatically, and they still demonstrate a 40% reduction compared with the count observed in the UNF code. As shown in Fig. 4.15, the UBER curves of GD/TC codes are still close in the early waterfall region like they are in the NLM channel simulations; however, they start to deviate at RBER less than 0.014. The TC code has no observed error floor in its performance curve as expected, and it has a 2 orders of

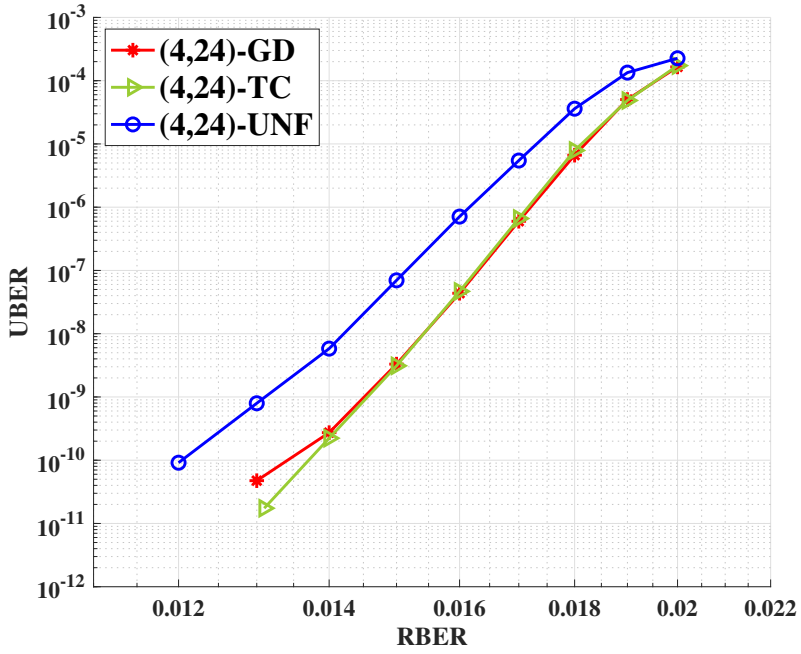


Figure 4.15: UBER curves of GD/TC/UNF codes with $(\gamma, \kappa) = (4, 24)$ in the BSC.

magnitude gain over the UNF code at RBER 0.013. The GD code curve surprisingly floors despite that there are no ASs with weight less than 8 in it: error profile analysis shows that the error floor at RBER 0.013 results from only 2 different large weight errors (one of weight 78 and another of weight 168) instead of structured small weight errors.

Remark 14. *While the reason why the large weight errors observed in the error profile of the GD code are detrimental in the BSC simulations remains unexplored and is left for future investigation, the TC code is observed to be robust against these errors, which is specifically intriguing. Moreover, the fact that the waterfall performance of GD/TC codes is remarkably superior to that of UNF codes calls for an asymptotic analysis that takes edge distribution into consideration. The significant gain achieved by TC codes substantiates the potential of TC codes in Flash memories.*

Fig. 4.16 shows FER curves of the GD/TC/UNF codes with $(\gamma, \kappa) = (4, 20)$ on the MR channel. The MR system adopts the partial-response (PR) channel presented in [82] and sequence detection. This PR channel incorporates the MR channel effects: inter-symbol

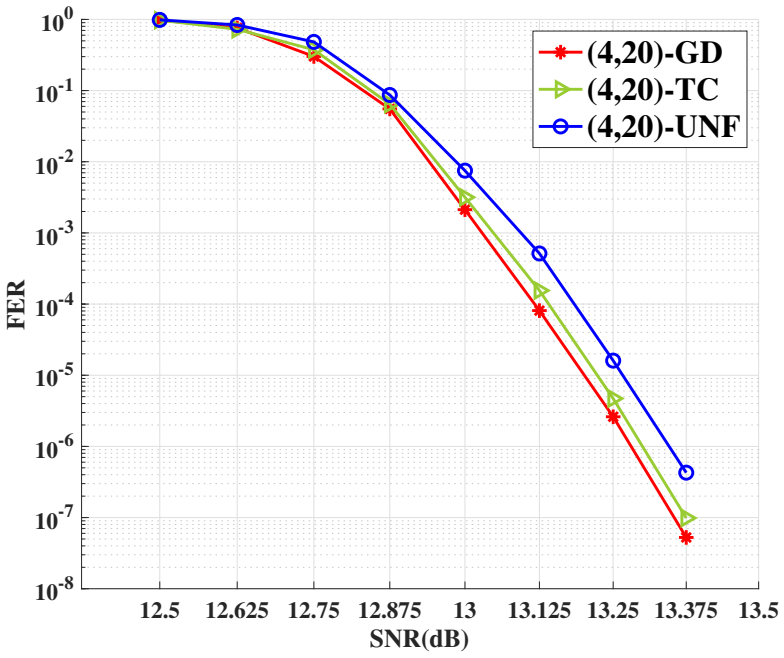


Figure 4.16: FER curves of GD/TC/UNF codes with $(\gamma, \kappa) = (4, 20)$ in the MR channel.

interference (intrinsic memory), jitter, and electronic noise. The normalized channel density [88, 82] is 1.4, and the PR equalization target is $(8, 14, 2)$. The filtering units are followed by a Bahl-Cocke-Jelinek-Raviv (BCJR) detector [89], which is based on pattern-dependent noise prediction (PDNP) [90], and again an FP FFT-QSPA ($q = 2$) LDPC decoder [74]. The number of global (detect-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 [82].

The partitioning matrices and the lifting matrices of the codes are specified in Section 4.6.3. The number of targeted objects (concatenated-cycle pairs) observed in the GD code demonstrates an approximate 40% reduction from the count observed in the UNF code. The FER of the GD/TC codes decreases with a rate that is approximately 13 orders of magnitude per 1 dB SNR increase. Moreover, the GD code has a significant gain of about 1 dB over the UNF code at SNR 13.375 dB. These results substantiate the remarkable impact of the GRADE-AO method in constructing SC codes with superior performance for stor-

age devices, with potential usage in further applications including wireless communication systems.

4.6.3 List of Partitioning Matrices and Lifting Matrices

Codes Simulated on the AWGN Channel (Fig. 4.11)

In this subsection, we specify partitioning matrices and lifting matrices for codes with $\gamma = 3$ simulated on the AWGN channel.

$$\mathbf{P}_{\text{GD}} = \begin{bmatrix} 2 & 0 & 5 & 5 & 0 & 0 & 4 \\ 5 & 5 & 0 & 0 & 1 & 1 & 0 \\ 0 & 2 & 4 & 5 & 5 & 5 & 0 \end{bmatrix}, \mathbf{L}_{\text{GD}} = \begin{bmatrix} 3 & 12 & 3 & 2 & 0 & 4 & 9 \\ 5 & 2 & 4 & 6 & 8 & 10 & 12 \\ 0 & 8 & 3 & 11 & 6 & 1 & 9 \end{bmatrix}. \tag{4.23}$$

$$\mathbf{P}_{\text{UNF}} = \begin{bmatrix} 1 & 3 & 0 & 3 & 2 & 5 & 5 \\ 4 & 0 & 2 & 0 & 5 & 3 & 1 \\ 0 & 4 & 2 & 1 & 4 & 2 & 1 \end{bmatrix}, \mathbf{L}_{\text{UNF}} = \begin{bmatrix} 2 & 5 & 6 & 4 & 0 & 3 & 0 \\ 9 & 2 & 8 & 11 & 5 & 10 & 12 \\ 0 & 8 & 3 & 11 & 1 & 1 & 9 \end{bmatrix}. \tag{4.24}$$

$$\begin{aligned} \mathbf{P}_{\text{GD}} &= \begin{bmatrix} 7 & 8 & 0 & 9 & 9 & 9 & 7 & 0 & 0 & 2 & 9 & 9 & 0 & 3 & 3 & 7 & 3 \\ 9 & 5 & 9 & 1 & 0 & 0 & 9 & 6 & 4 & 7 & 2 & 2 & 4 & 0 & 0 & 9 & 8 \\ 1 & 1 & 8 & 0 & 8 & 8 & 0 & 9 & 9 & 6 & 0 & 0 & 9 & 9 & 9 & 0 & 0 \end{bmatrix}. \\ \mathbf{L}_{\text{GD}} &= \begin{bmatrix} 0 & 6 & 0 & 3 & 0 & 6 & 0 & 6 & 6 & 6 & 4 & 2 & 0 & 0 & 6 & 0 & 0 \\ 3 & 2 & 2 & 6 & 2 & 3 & 4 & 0 & 0 & 0 & 6 & 1 & 2 & 5 & 0 & 1 & 4 \\ 5 & 1 & 0 & 3 & 6 & 6 & 6 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 0 & 1 & 2 \end{bmatrix}. \end{aligned} \tag{4.25}$$

$$\mathbf{P}_{\text{UNF}} = \begin{bmatrix} 9 & 2 & 9 & 4 & 8 & 8 & 4 & 6 & 0 & 3 & 9 & 3 & 1 & 2 & 5 & 8 & 3 \\ 3 & 9 & 8 & 1 & 0 & 8 & 4 & 0 & 5 & 7 & 7 & 0 & 4 & 9 & 5 & 0 & 3 \\ 5 & 6 & 2 & 9 & 1 & 0 & 7 & 2 & 7 & 1 & 2 & 7 & 6 & 0 & 8 & 1 & 6 \end{bmatrix}.$$

$$\mathbf{L}_{\text{UNF}} = \begin{bmatrix} 0 & 3 & 2 & 2 & 2 & 0 & 3 & 0 & 2 & 1 & 0 & 1 & 2 & 0 & 0 & 0 & 0 \\ 4 & 2 & 4 & 6 & 1 & 1 & 2 & 1 & 0 & 6 & 1 & 1 & 3 & 3 & 0 & 1 & 4 \\ 4 & 1 & 2 & 1 & 4 & 5 & 6 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 0 & 1 & 2 \end{bmatrix}. \tag{4.26}$$

Codes Simulated on the AWGN Channel (Fig. 4.12)

In this subsection, we specify partitioning matrices and lifting matrices for codes with $(\gamma, \kappa, m, z, L) = (4, 29, 19, 29, 20)$ simulated on the AWGN channel.

$\mathbf{P}_{\text{GD}} =$

$$\begin{bmatrix} 0 & 0 & 0 & 19 & 17 & 17 & 1 & 11 & 13 & 5 & 18 & 10 & 19 & 13 & 1 & 6 & 19 & 8 & 19 & 0 & 19 & 0 & 0 & 0 & 2 & 17 & 6 & 19 & 4 \\ 19 & 18 & 18 & 2 & 0 & 19 & 19 & 5 & 3 & 19 & 9 & 2 & 9 & 9 & 3 & 17 & 6 & 0 & 2 & 16 & 12 & 13 & 8 & 18 & 16 & 0 & 17 & 10 & 0 \\ 1 & 14 & 3 & 16 & 7 & 1 & 4 & 19 & 5 & 0 & 0 & 16 & 0 & 0 & 7 & 19 & 10 & 19 & 16 & 18 & 3 & 18 & 15 & 3 & 19 & 8 & 19 & 1 & 15 \\ 16 & 0 & 14 & 1 & 11 & 2 & 15 & 2 & 19 & 16 & 18 & 0 & 19 & 19 & 19 & 0 & 0 & 5 & 1 & 0 & 9 & 4 & 19 & 14 & 7 & 12 & 0 & 19 & 1 \end{bmatrix}.$$

$\mathbf{L}_{\text{GD}} =$

$$\begin{bmatrix} 7 & 1 & 18 & 21 & 5 & 4 & 17 & 0 & 6 & 16 & 26 & 8 & 13 & 7 & 5 & 6 & 9 & 2 & 0 & 0 & 0 & 0 & 0 & 5 & 0 & 4 & 19 & 0 & 0 \\ 3 & 15 & 4 & 1 & 5 & 3 & 12 & 19 & 10 & 21 & 19 & 3 & 4 & 19 & 28 & 1 & 3 & 5 & 12 & 18 & 11 & 10 & 15 & 17 & 19 & 21 & 18 & 25 & 27 \\ 0 & 8 & 16 & 24 & 3 & 11 & 20 & 20 & 6 & 14 & 22 & 1 & 9 & 2 & 25 & 21 & 12 & 7 & 28 & 6 & 15 & 23 & 19 & 10 & 0 & 1 & 4 & 13 & 21 \\ 0 & 18 & 7 & 25 & 1 & 3 & 21 & 10 & 28 & 17 & 6 & 24 & 13 & 2 & 20 & 9 & 27 & 16 & 5 & 23 & 12 & 1 & 19 & 8 & 26 & 15 & 4 & 22 & 11 \end{bmatrix}.$$

(4.27)

$$\mathbf{P}_{\text{UNF}} =$$

$$\begin{bmatrix}
0 & 17 & 3 & 13 & 1 & 14 & 4 & 12 & 4 & 15 & 10 & 2 & 17 & 2 & 18 & 11 & 17 & 15 & 11 & 3 & 13 & 12 & 13 & 6 & 2 & 5 & 14 & 13 & 14 \\
8 & 0 & 19 & 19 & 18 & 8 & 5 & 18 & 13 & 6 & 11 & 3 & 2 & 4 & 11 & 3 & 9 & 15 & 16 & 7 & 7 & 12 & 19 & 16 & 4 & 9 & 0 & 13 & 3 \\
14 & 6 & 12 & 10 & 12 & 1 & 17 & 9 & 7 & 5 & 16 & 19 & 1 & 15 & 5 & 19 & 6 & 5 & 15 & 7 & 0 & 2 & 3 & 10 & 15 & 9 & 6 & 7 & 11 \\
17 & 14 & 0 & 2 & 9 & 18 & 12 & 1 & 8 & 11 & 4 & 4 & 7 & 10 & 1 & 8 & 14 & 8 & 0 & 16 & 17 & 16 & 1 & 0 & 10 & 18 & 18 & 10 & 8
\end{bmatrix}.$$

$$\mathbf{L}_{\text{UNF}} =$$

$$\begin{bmatrix}
12 & 1 & 1 & 7 & 14 & 27 & 4 & 26 & 25 & 2 & 0 & 6 & 15 & 7 & 24 & 1 & 1 & 6 & 17 & 5 & 13 & 19 & 2 & 0 & 11 & 0 & 0 & 0 & 1 \\
5 & 2 & 4 & 22 & 8 & 5 & 23 & 1 & 4 & 18 & 28 & 1 & 19 & 17 & 22 & 6 & 3 & 3 & 14 & 9 & 11 & 13 & 15 & 3 & 0 & 2 & 3 & 25 & 27 \\
23 & 8 & 2 & 24 & 3 & 7 & 1 & 27 & 6 & 14 & 21 & 12 & 9 & 17 & 5 & 4 & 12 & 20 & 28 & 7 & 7 & 13 & 2 & 25 & 18 & 26 & 5 & 13 & 21 \\
28 & 18 & 7 & 4 & 14 & 3 & 21 & 10 & 28 & 17 & 6 & 24 & 13 & 2 & 9 & 8 & 1 & 26 & 5 & 23 & 12 & 1 & 19 & 8 & 26 & 15 & 4 & 22 & 11
\end{bmatrix}. \tag{4.28}$$

Codes Simulated on the AWGN Channel (Fig. 4.13)

In this subsection, we specify partitioning matrices and lifting matrices for codes with $(\gamma, \kappa) = (4, 17)$ simulated on the AWGN channel.

$$\begin{aligned}
\mathbf{P}_{\text{TC}} &= \begin{bmatrix}
4 & 4 & 4 & 4 & 4 & 0 & 0 & 0 & 1 & 1 & 4 & 1 & 1 & 1 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 4 & 4 & 4 & 4 & 4 & 1 \\
0 & 0 & 1 & 1 & 1 & 4 & 4 & 4 & 4 & 4 & 1 & 0 & 0 & 0 & 1 & 1 & 4 \\
1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 4 & 4 & 4 & 4 & 4
\end{bmatrix}. \\
\mathbf{L}_{\text{TC}} &= \begin{bmatrix}
16 & 7 & 5 & 15 & 0 & 2 & 9 & 2 & 1 & 3 & 2 & 7 & 0 & 12 & 7 & 13 & 13 \\
0 & 6 & 4 & 0 & 10 & 10 & 12 & 14 & 9 & 1 & 11 & 2 & 7 & 1 & 16 & 4 & 13 \\
1 & 8 & 16 & 7 & 15 & 6 & 14 & 5 & 13 & 3 & 5 & 3 & 9 & 11 & 7 & 0 & 11 \\
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 1 & 11 & 9 & 5 & 4 & 15 & 16
\end{bmatrix}.
\end{aligned} \tag{4.29}$$

$$\mathbf{P}_{\mathrm{SC}} = \begin{bmatrix} 2 & 2 & 2 & 2 & 2 & 0 & 0 & 0 & 1 & 1 & 2 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 2 & 2 & 2 & 2 & 2 & 2 \\ 0 & 0 & 1 & 1 & 1 & 2 & 2 & 2 & 2 & 2 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 2 & 2 & 2 & 2 \end{bmatrix}.$$

$$\mathbf{L}_{\mathrm{SC},z=17} = \begin{bmatrix} 5 & 5 & 12 & 1 & 7 & 16 & 10 & 0 & 0 & 0 & 14 & 15 & 1 & 10 & 7 & 0 & 10 \\ 0 & 2 & 15 & 0 & 2 & 9 & 12 & 14 & 16 & 1 & 3 & 5 & 7 & 9 & 11 & 13 & 15 \\ 1 & 8 & 16 & 7 & 15 & 6 & 16 & 5 & 13 & 4 & 12 & 3 & 12 & 2 & 10 & 1 & 9 \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 \end{bmatrix}. \tag{4.30}$$

$$\mathbf{L}_{\mathrm{SC},z=28} = \begin{bmatrix} 21 & 27 & 3 & 13 & 1 & 5 & 27 & 16 & 21 & 4 & 0 & 15 & 26 & 2 & 11 & 8 & 0 \\ 15 & 11 & 1 & 20 & 22 & 11 & 21 & 20 & 3 & 0 & 22 & 22 & 24 & 26 & 0 & 2 & 4 \\ 25 & 13 & 22 & 24 & 9 & 12 & 15 & 0 & 18 & 16 & 24 & 21 & 12 & 20 & 0 & 8 & 18 \\ 0 & 18 & 8 & 26 & 16 & 17 & 24 & 14 & 1 & 22 & 12 & 2 & 11 & 10 & 0 & 18 & 8 \end{bmatrix}.$$

Codes Simulated on the NLM Channel

In this subsection, we specify partitioning matrices and lifting matrices for codes with $(\gamma, \kappa, m, z, L) = (4, 24, 6, 17, 40)$ simulated on the NLM channel.

$$\mathbf{P}_{\mathrm{GD}} = \begin{bmatrix} 0 & 6 & 2 & 0 & 6 & 1 & 0 & 1 & 6 & 6 & 6 & 6 & 0 & 0 & 0 & 4 & 5 & 4 & 5 & 0 & 0 & 5 & 0 & 1 \\ 0 & 0 & 6 & 2 & 6 & 6 & 6 & 0 & 1 & 1 & 5 & 6 & 4 & 6 & 0 & 0 & 1 & 6 & 6 & 0 & 6 & 1 & 4 & 0 \\ 3 & 3 & 6 & 6 & 0 & 6 & 1 & 5 & 3 & 0 & 0 & 1 & 2 & 6 & 6 & 6 & 2 & 0 & 0 & 6 & 4 & 6 & 0 & 6 \\ 6 & 5 & 1 & 5 & 2 & 0 & 3 & 5 & 0 & 3 & 0 & 0 & 6 & 0 & 6 & 3 & 6 & 2 & 0 & 6 & 0 & 0 & 6 & 4 \end{bmatrix}.$$

$$\mathbf{L}_{\mathrm{GD}} = \begin{bmatrix} 16 & 15 & 1 & 7 & 8 & 11 & 14 & 1 & 5 & 6 & 16 & 9 & 12 & 0 & 5 & 13 & 1 & 0 & 3 & 5 & 15 & 0 & 0 & 1 \\ 9 & 2 & 8 & 6 & 8 & 10 & 10 & 3 & 5 & 8 & 5 & 5 & 7 & 12 & 9 & 14 & 15 & 0 & 2 & 4 & 6 & 8 & 10 & 12 \\ 0 & 2 & 6 & 7 & 15 & 12 & 4 & 5 & 13 & 4 & 7 & 8 & 11 & 2 & 1 & 1 & 8 & 0 & 8 & 16 & 9 & 15 & 11 & 5 \\ 0 & 1 & 2 & 3 & 3 & 5 & 6 & 7 & 14 & 8 & 14 & 11 & 12 & 13 & 14 & 15 & 16 & 0 & 1 & 2 & 3 & 4 & 11 & 6 \end{bmatrix}. \tag{4.31}$$

$$\mathbf{P}_{\mathrm{TC}} = \begin{bmatrix} 0 & 1 & 4 & 6 & 6 & 1 & 0 & 1 & 4 & 6 & 4 & 6 & 6 & 6 & 0 & 6 & 1 & 1 & 1 & 0 & 4 & 0 & 1 & 0 \\ 1 & 6 & 0 & 6 & 4 & 6 & 6 & 0 & 1 & 6 & 1 & 6 & 0 & 0 & 6 & 0 & 0 & 6 & 6 & 4 & 0 & 1 & 0 & 4 \\ 6 & 6 & 6 & 0 & 0 & 0 & 1 & 4 & 0 & 0 & 6 & 1 & 1 & 4 & 6 & 4 & 6 & 4 & 0 & 4 & 6 & 6 & 4 & 0 \\ 4 & 0 & 1 & 1 & 1 & 6 & 4 & 6 & 6 & 0 & 0 & 1 & 4 & 4 & 0 & 1 & 6 & 0 & 4 & 6 & 4 & 4 & 6 & 6 \end{bmatrix}.$$

$$\mathbf{L}_{\mathrm{TC}} = \begin{bmatrix} 13 & 4 & 1 & 7 & 1 & 11 & 3 & 6 & 6 & 6 & 1 & 12 & 15 & 14 & 5 & 4 & 0 & 12 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 11 & 4 & 9 & 13 & 15 & 0 & 14 & 11 & 1 & 7 & 8 & 7 & 9 & 9 & 12 & 2 & 0 & 2 & 4 & 6 & 14 & 4 & 16 \\ 0 & 8 & 16 & 1 & 15 & 6 & 14 & 5 & 14 & 4 & 9 & 3 & 11 & 2 & 10 & 1 & 9 & 0 & 8 & 13 & 12 & 15 & 6 & 11 \\ 0 & 5 & 2 & 3 & 4 & 5 & 6 & 2 & 8 & 9 & 8 & 4 & 12 & 13 & 14 & 15 & 0 & 9 & 15 & 13 & 3 & 2 & 8 & 4 \end{bmatrix}. \tag{4.32}$$

$$\mathbf{P}_{\mathrm{UNF}} = \begin{bmatrix} 0 & 5 & 6 & 5 & 0 & 6 & 2 & 2 & 5 & 1 & 2 & 6 & 2 & 0 & 6 & 2 & 2 & 3 & 2 & 3 & 0 & 3 & 1 & 1 \\ 5 & 4 & 0 & 5 & 0 & 2 & 3 & 1 & 0 & 0 & 4 & 0 & 1 & 4 & 3 & 6 & 2 & 4 & 6 & 4 & 6 & 6 & 4 & 3 \\ 1 & 1 & 5 & 0 & 6 & 4 & 6 & 5 & 1 & 5 & 1 & 4 & 2 & 5 & 0 & 3 & 1 & 3 & 2 & 1 & 5 & 3 & 4 & 3 \\ 6 & 1 & 2 & 2 & 4 & 4 & 1 & 5 & 6 & 6 & 3 & 1 & 5 & 0 & 0 & 4 & 5 & 1 & 4 & 3 & 0 & 0 & 3 & 6 \end{bmatrix}.$$

$$\mathbf{L}_{\mathrm{UNF}} = \begin{bmatrix} 8 & 16 & 7 & 7 & 0 & 9 & 6 & 14 & 6 & 14 & 1 & 13 & 14 & 8 & 0 & 0 & 4 & 6 & 0 & 0 & 0 & 0 & 3 & 16 \\ 3 & 2 & 6 & 6 & 13 & 10 & 12 & 14 & 10 & 3 & 3 & 11 & 7 & 9 & 11 & 7 & 15 & 0 & 13 & 4 & 4 & 8 & 10 & 12 \\ 2 & 8 & 16 & 1 & 15 & 6 & 14 & 5 & 13 & 2 & 12 & 3 & 11 & 2 & 13 & 1 & 9 & 0 & 8 & 16 & 7 & 15 & 6 & 14 \\ 1 & 1 & 2 & 3 & 13 & 5 & 11 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 0 & 1 & 2 & 3 & 4 & 5 & 6 \end{bmatrix}. \tag{4.33}$$

Codes Simulated on the BSC

In this subsection, we specify the new lifting matrices adopted for GD/TC codes with $(\gamma, \kappa, m, z, L) = (4, 24, 6, 17, 40)$ simulated on the BSC.

$$\mathbf{L}_{\mathrm{GD}} = \begin{bmatrix} 11 & 15 & 1 & 7 & 8 & 11 & 14 & 1 & 5 & 6 & 16 & 9 & 12 & 0 & 5 & 13 & 1 & 0 & 3 & 5 & 15 & 0 & 0 & 1 \\ 9 & 2 & 8 & 6 & 8 & 10 & 10 & 3 & 5 & 8 & 5 & 5 & 7 & 12 & 9 & 14 & 15 & 0 & 2 & 4 & 6 & 8 & 10 & 12 \\ 0 & 2 & 6 & 7 & 15 & 12 & 4 & 5 & 13 & 4 & 7 & 8 & 11 & 2 & 1 & 1 & 8 & 0 & 8 & 16 & 9 & 15 & 11 & 5 \\ 0 & 1 & 2 & 3 & 3 & 5 & 6 & 7 & 14 & 8 & 14 & 11 & 12 & 13 & 14 & 15 & 16 & 0 & 1 & 2 & 3 & 4 & 11 & 6 \end{bmatrix}. \tag{4.34}$$

$$\mathbf{L}_{\mathrm{TC}} = \begin{bmatrix} 13 & 4 & 1 & 7 & 1 & 11 & 3 & 6 & 6 & 6 & 1 & 12 & 15 & 14 & 5 & 4 & 0 & 12 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 11 & 4 & 9 & 13 & 15 & 0 & 14 & 11 & 1 & 7 & 8 & 7 & 9 & 9 & 12 & 2 & 0 & 2 & 4 & 6 & 14 & 4 & 16 \\ 0 & 8 & 16 & 1 & 15 & 6 & 14 & 5 & 14 & 4 & 9 & 3 & 11 & 2 & 10 & 1 & 9 & 0 & 8 & 13 & 12 & 15 & 6 & 11 \\ 0 & 9 & 2 & 3 & 4 & 5 & 6 & 2 & 8 & 9 & 8 & 4 & 12 & 13 & 14 & 15 & 0 & 9 & 15 & 13 & 3 & 2 & 8 & 4 \end{bmatrix}. \tag{4.35}$$

Codes Simulated on the MR channel

In this subsection, we specify partitioning matrices and lifting matrices for codes with $(\gamma, \kappa, m, z, L) = (4, 20, 6, 13, 20)$ simulated on the MR channel.

$$\begin{aligned} \mathbf{P}_{\mathrm{GD}} &= \begin{bmatrix} 0 & 0 & 3 & 6 & 0 & 1 & 2 & 2 & 0 & 6 & 6 & 0 & 6 & 5 & 6 & 0 & 6 & 0 & 5 & 6 \\ 3 & 0 & 1 & 0 & 5 & 0 & 6 & 3 & 2 & 0 & 6 & 3 & 1 & 6 & 6 & 6 & 0 & 6 & 6 & 5 \\ 6 & 6 & 0 & 1 & 1 & 6 & 1 & 6 & 5 & 0 & 0 & 4 & 5 & 0 & 0 & 4 & 0 & 5 & 2 & 3 \\ 1 & 6 & 6 & 4 & 6 & 4 & 6 & 0 & 6 & 4 & 0 & 6 & 0 & 1 & 0 & 0 & 6 & 2 & 0 & 0 \end{bmatrix}. \\ \mathbf{L}_{\mathrm{GD}} &= \begin{bmatrix} 3 & 5 & 4 & 8 & 3 & 11 & 0 & 2 & 0 & 1 & 0 & 9 & 6 & 9 & 0 & 0 & 2 & 2 & 1 & 1 \\ 7 & 2 & 6 & 6 & 8 & 4 & 8 & 1 & 2 & 6 & 6 & 4 & 6 & 7 & 7 & 7 & 6 & 8 & 5 & 10 \\ 0 & 7 & 3 & 11 & 0 & 7 & 9 & 1 & 12 & 0 & 2 & 10 & 5 & 0 & 8 & 3 & 11 & 7 & 1 & 9 \\ 0 & 5 & 10 & 2 & 7 & 12 & 4 & 11 & 1 & 6 & 11 & 5 & 8 & 7 & 5 & 10 & 2 & 7 & 12 & 4 \end{bmatrix}. \end{aligned} \tag{4.36}$$

$$\mathbf{P}_{\mathrm{TC}} = \begin{bmatrix} 6 & 0 & 0 & 6 & 0 & 0 & 4 & 6 & 0 & 1 & 6 & 6 & 6 & 1 & 6 & 6 & 0 & 0 & 1 & 1 \\ 1 & 6 & 4 & 4 & 4 & 1 & 1 & 6 & 4 & 6 & 0 & 0 & 0 & 6 & 0 & 4 & 6 & 0 & 4 & 6 \\ 4 & 0 & 6 & 4 & 1 & 6 & 1 & 1 & 4 & 4 & 4 & 6 & 0 & 0 & 1 & 0 & 1 & 6 & 6 & 4 \\ 0 & 6 & 4 & 0 & 6 & 4 & 6 & 0 & 6 & 0 & 1 & 1 & 4 & 4 & 6 & 1 & 6 & 6 & 0 & 0 \end{bmatrix}.$$

$$\mathbf{L}_{\mathrm{TC}} = \begin{bmatrix} 1 & 2 & 8 & 5 & 11 & 3 & 0 & 1 & 10 & 4 & 1 & 12 & 10 & 0 & 10 & 3 & 0 & 6 & 0 & 0 \\ 8 & 7 & 4 & 12 & 5 & 10 & 12 & 1 & 3 & 5 & 10 & 10 & 4 & 8 & 2 & 7 & 6 & 12 & 10 & 12 \\ 0 & 11 & 3 & 11 & 0 & 1 & 9 & 4 & 12 & 7 & 8 & 4 & 9 & 10 & 8 & 3 & 11 & 2 & 1 & 10 \\ 0 & 12 & 10 & 2 & 7 & 12 & 4 & 9 & 1 & 6 & 4 & 10 & 11 & 0 & 5 & 9 & 2 & 7 & 12 & 4 \end{bmatrix}. \tag{4.37}$$

$$\mathbf{P}_{\mathrm{UNF}} = \begin{bmatrix} 0 & 0 & 6 & 5 & 5 & 1 & 5 & 1 & 6 & 3 & 3 & 2 & 2 & 3 & 2 & 6 & 0 & 3 & 0 & 2 \\ 5 & 5 & 2 & 1 & 1 & 2 & 5 & 2 & 0 & 3 & 5 & 3 & 5 & 5 & 1 & 6 & 6 & 1 & 2 & 2 \\ 6 & 0 & 1 & 5 & 2 & 4 & 1 & 4 & 4 & 0 & 4 & 4 & 6 & 0 & 2 & 0 & 6 & 3 & 4 & 4 \\ 0 & 4 & 0 & 3 & 4 & 5 & 1 & 5 & 1 & 6 & 1 & 0 & 1 & 4 & 6 & 3 & 0 & 6 & 3 & 6 \end{bmatrix}.$$

$$\mathbf{L}_{\mathrm{UNF}} = \begin{bmatrix} 6 & 11 & 1 & 1 & 4 & 9 & 11 & 0 & 3 & 11 & 12 & 0 & 0 & 1 & 0 & 9 & 2 & 0 & 10 & 1 \\ 10 & 3 & 4 & 5 & 9 & 2 & 2 & 3 & 8 & 5 & 7 & 9 & 5 & 12 & 0 & 4 & 6 & 8 & 10 & 12 \\ 12 & 0 & 3 & 11 & 5 & 1 & 9 & 12 & 12 & 7 & 4 & 4 & 5 & 12 & 3 & 7 & 11 & 6 & 0 & 9 \\ 9 & 5 & 10 & 2 & 7 & 10 & 4 & 9 & 1 & 6 & 11 & 5 & 8 & 5 & 5 & 11 & 9 & 7 & 12 & 4 \end{bmatrix}. \tag{4.38}$$

4.7 Conclusion

Discrete optimization of the constructions of spatially-coupled (SC) codes with high memories is known to be computationally expensive. Heuristic algorithms are efficient, but can hardly guarantee the performance because of the lack of theoretical guidance. In this chapter, we proposed the so-called GRADE-AO method, a probabilistic framework that efficiently searches for locally optimal QC-SC codes with arbitrary memories. We obtained a locally optimal edge distribution that minimizes the expected number of the most detrimental objects via gradient descent. Starting from a random partitioning matrix with the derived edge distribution, we then applied a semi-greedy algorithm to find a locally optimal partitioning

matrix near it. While the application of GRADE-AO in optimizing the number of short cycles has shown noticeable gains, we focused in this chapter on minimizing the number of more detrimental objects, the concatenated cycles. This finer-grained optimization avoids unnecessary attention on individual cycles which are typically not problematic on their own, especially in codes with high VN degrees and irregular codes. Simulation results show that our proposed constructions have a significant performance gain over state-of-the-art codes; this gain is shown to be universal in both waterfall and error floor regions, as well as on channels underlying various practical systems. Future work includes extending the framework to other classes of underlying block codes.

CHAPTER 5

SC-IRA Codes for Quantum Key Distribution

5.1 Introduction

Quantum key distribution (QKD), known for offering a physically secure way to effectively share private keys that are physically random over a quantum communication channel observed by an eavesdropper, has been intensely investigated [48, 49, 50, 51, 52, 53, 91, 92]. The ultimate goal of QKD is to achieve a high secure key rate (SKR) within the repeaterless limit, namely, to extract the highest number of bits per generated entangled photon pairs [93]. Entanglement-based protocols have thus received significant attention due to their ability to enable dense communications in high-noise environments. Furthermore, they have a tremendous potential to be scaled up to multi-party networks with high security [91, 92, 54, 55]. Various entanglement bases can be adopted, including orbital angular momentum, spatial and polarization, and energy-time basis. Energy-time entanglement-based protocols are of particular interest owing to their ability to deliver high key rates through high-dimensional Hilbert space [91, 94, 95] and for offering provable security. While finite key effects and security proofs against collective attacks on energy-time protocols have been well established [96, 97, 98, 99], the energy-time entanglement based protocols are potentially able to further achieve unconditional security through non-local Franson and conjugate-Franson interferometry [92], which is critical in secure communications. Despite all their advantages, photonic quantum communication inevitably happens under "photon-starved" conditions due to low

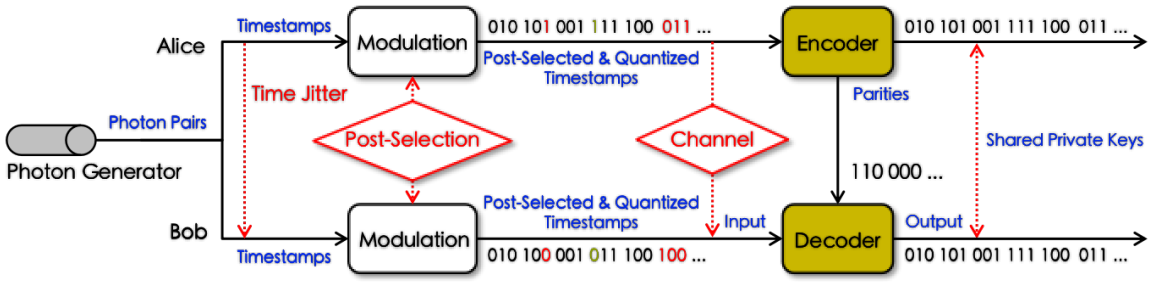


Figure 5.1: Block diagram of a time-bin QKD protocol. There are 3 steps: 1) Entangled photon pairs are generated and sent to Alice and Bob; 2) Timestamps of the sequences are quantized and post-selected into multi-bit data streams; 3) Error correction codes (ECCs) are adopted to extract a common sequence from the post-selected data streams; this sequence is used as the private key for the secure quantum communication.

state-of-the-art source/detector rates, high photon losses, noise, and lack of quantum repeater devices that could extend the range of coherent qubit transfer [50, 93]. The benefits of moving towards high-dimensionality also come at the cost of increased probability of symbol errors, resultant in a potential degradation of photon utilization.

Fig. 5.1 presents a block diagram of the major steps in an energy-time QKD protocol. The entangled photon streams received by Alice and Bob are inevitably nonidentical due to a variety of practical issues, including timing jitters, photon losses, and dark counts, all of which reduce the secure key rate communicated over entangled photon pairs. Error correction coding (ECC) has been the major mathematical tool [91, 92, 54, 55] for information reconciliation to overcome the aforementioned transmission noise and extract the mutual information (private key) to be shared by Alice and Bob. There are two central parameters associated with the SKR. The first parameter is the fraction of the selected photon pairs out of the total number of generated photon pairs. This fraction is referred to as the raw photon efficiency (RPE), which is dependent on the modulation scheme. Second parameter is the number of qubits carried per selected photon pairs after the encoding and decoding. This number is referred to as the photon information efficiency (PIE), which is dependent on the channel and the adopted ECC scheme. Strategically tuning the discretization bin widths to include more frames containing accidental concurrent detections effectively leads to desirable

higher RPE [91]. The adaptive modulation scheme proposed in [100] can further increase the RPE through adaptive frame sizes (number of bins per frame). However, the gain in RPE can be compromised by the resultant worsening channel, as it has a higher fraction of uniformly distributed errors. The presence of a large fraction of such errors prevents existing coding schemes such as the MLC scheme presented in [5] from sustaining a high reconciliation efficiency achieved in the low RPE scenario. In this work, we propose ECC solutions that are exceedingly more robust than existing schemes in combating uniform errors in the context of the energy-time QKD transmissions.

One major coding framework considered for time-bin protocols is the "multi-level coding, multi-stage decoding" scheme proposed by Zhou *et al.* [5], referred to later as the MLC scheme. MLC encodes and decodes the bitstream at each bit layer serially and utilizes the dependencies between bits from different layers in the same symbol. MLC scheme has been considered for multiple works [91, 101]. Advanced coding schemes including SC codes, IRA codes, and the more general class of multi-edge-type (MET) codes that subsumes IRA codes have been investigated in the continuous variable (CV) QKD setting [54, 55]. SC-IRA codes have also been studied in [56], also under the CVQKD setting. However, codes for CVQKD are discussed under the binary input additive white Gaussian noise (BIAWGN) channel, which is completely different from the (non-binary) channel observed in the time-bin protocols. Therefore, coding solutions designed for CVQKD are not able to be used in time-bin protocols focused in this paper.

While the MLC framework is effective in combating highly asymmetric errors resultant from jitter errors, the framework is highly reliant on the dependency between the bit layers. It is thus vulnerable to uniformly distributed errors resultant from accidental coincidences. Unfortunately, modulation schemes with higher RPE inevitably lead to a larger number of accidental coincidences after post-selection, which calls for ECCs that are more robust to uniformly distributed errors to reach a better photon utilization. Unlike the MLC coding, which is by design binary, non-binary (NB) low-density-parity-check (LDPC) codes are not

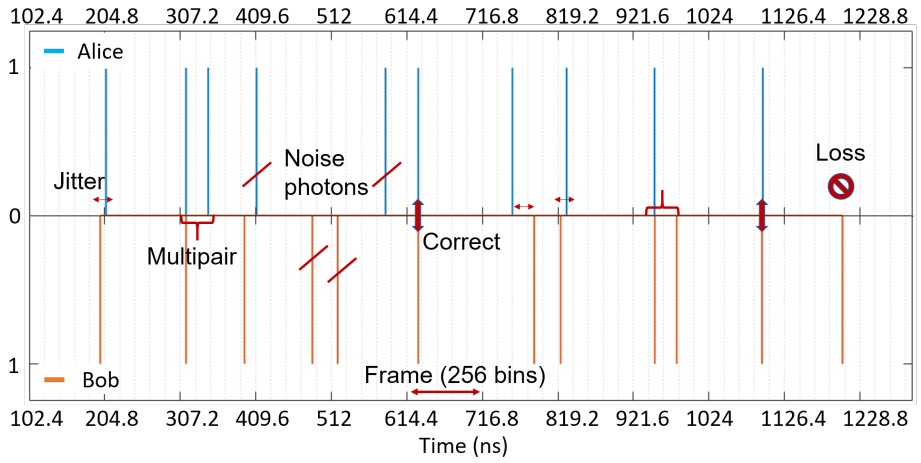


Figure 5.2: Photon arrival time at Alice and Bob. Arrival timestamps of photons are quantized into bins with a bin width 400ps and consecutive 256 (can be any other powers of 2) bins are grouped into frames. Frames containing multi-pairs and losses are discarded such that each selected frame has exactly one photon from Alice and from Bob. Carefully tuning the bin width to include more accidental concurrent detections effectively leads to higher RPE, which is nonetheless at the cost of having more uniformly distributed errors, resulting in a lower PIE.

reliant on the dependencies between different bit layers. In this work, we develop NB-LDPC codes tailored to be more robust than the state-of-the-art MLC scheme against errors incurred by modulations with high RPE.

In particular, we propose non-binary **spatially-coupled irregular-repeat-accumulate (SC-IRA) codes**. SC-IRA codes are high-performance codes that strategically combine two advanced LDPC codes, the spatially-coupled (SC) codes and the irregular-repeat-accumulate (IRA) codes. Building upon the optimization method proposed in [57], we further present a framework that systematically constructs high-performance finite-length SC-IRA codes. Our work Simulation results on experimental data demonstrate the superiority of our codes relative to the MLC coding scheme, especially in the regime where RPE is higher.

5.2 System Model

5.2.1 Major Steps and Related Measures

In this section, we first briefly introduce the QKD protocols of interest, the time-bin protocols, and the mathematical definitions of the related measures. In the modulation step shown in Fig. 5.1, the time domain is divided into non-overlapping frames, and each frame consists of 2^M, $M \geq 1$, bins of an identical length. Thus each arrival timestamp can be represented by an M-bit symbol from the Galois field GF(2^M). All the frames are subject to post-selection. In particular, only the frames where both Alice and Bob have exactly one detection are regarded as selected, and others are discarded. A raw data stream from our testbed is shown in Fig. 5.2, where various cases affecting reliable communications are annotated. The associated measure is the raw photon efficiency (RPE):

$$RPE = \frac{\text{Number of selected frames}}{\text{Total number of generated photon pairs}}. \tag{5.1}$$

Photons in a selected frame are not necessarily from an entangled photon pair but might also be two non-entangled photons that accidentally fall into the same frame. Different modulation schemes lead to a different number of accidental concurrent pairs, resulting in different RPEs. Fig. 5.3 presents the RPE under different combinations of bin widths and frame sizes.

In practical implementations, the two symbols carried by photons in a selected frame can be different due to timing jitters between entangled photons and the random offsets between accidental concurrent photon pairs. The sequences received by Alice and Bob are divided into groups (referred to as blocks) such that each of them consists of N (referred to as the blocksize or as codelength) symbols. Let $\mathbf{X}$, $\mathbf{Y} \in \text{GF}(2^M)^N$, denote blocks recorded by Alice and Bob, respectively. Then, $\mathbf{Y}$ is a noisy version of $\mathbf{X}$. Bob is able to retrieve $\mathbf{X}$ from $\mathbf{Y}$ and a parity sequence $\mathbf{HX}$ sent from Alice through ECCs by regarding $\mathbf{H} \in \text{GF}(2^M)^{K \times N}$ as a parity check matrix of some linear block code. The rate $R = M \times K/N$ (bits/photon) affects the PIE.

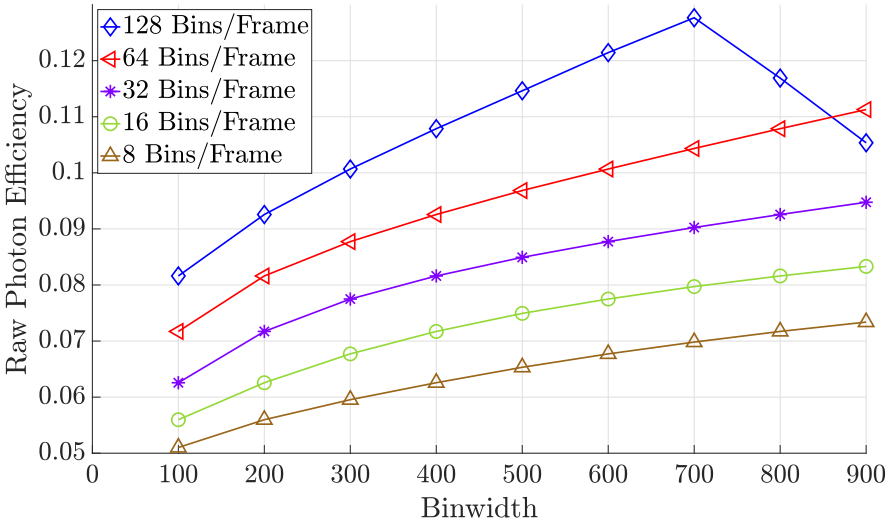


Figure 5.3: Raw photon efficiencies at different bin widths and frame sizes. The number of post-selected photon pairs increases with respect to the bin width, which results in an increased fraction of global errors due to the inclusion of background-related accidental concurrent detections.

In particular, PIE and SKR are defined as follows. The Holevo information γ_{BE} measures the amount of information an eavesdropper, commonly referred to as Eve, has on the secret keys. The reconciliation efficiency $\beta = \frac{R}{I_{AB}}$, where I_{AB} is the channel capacity of the QKD channel, specifies the theoretical upper limit of R. The number of photons being generated per second is denoted by N_0,

$$\begin{aligned} PIE &= \beta I_{AB} - \gamma_{BE}, \\ SKR &= PIE \times RPE \times N_0. \end{aligned} \tag{5.2}$$

When γ_{BE} is fixed, PIE is determined by the rate R. Considering the wasted frames due to frame errors, measured by frame error rate (FER), we use the normalized rate (NR) to measure the PIE, which is specified as follows:

$$NR = (1 - FER) \times R. \tag{5.3}$$

Note that the codes being constructed and compared with in this paper all result in FERs

that are small ($< 0.04\%$), thus NR is very close to FER in the codes mentioned here. The relation between SKR and NR is therefore stated as follows:

$$SKR \approx RPE \times (NR - \gamma_{BE}) \times N_0. \tag{5.4}$$

While it is easy to compute and measure NPE and NR, we next consider the normalized photon efficiency (NPE), i.e., the number of bits carried per generated photon pairs, as an alternative measure of SKR. Quantity NPE is defined as follows and it is demonstrated later on that the gain in NPE is a legitimate lower bound of the gain in SKR,

$$NPE = RPE \times NR. \tag{5.5}$$

Fix N_0. Consider two ECCs, referred to scheme 1 with NR_1, RPE_1, and SKR_1, and scheme 2 with NR_2, RPE_2, and SKR_2 ($NR_1 < NR_2$). Then, the gain of scheme 2 over scheme 1 is $\left(\frac{RPE_2(NR_2-\gamma_{BE})}{RPE_1(NR_1-\gamma_{BE})} - 1\right)$. This gain is strictly greater than the gain of NPE, $\left(\frac{RPE_2 \times NR_2}{RPE_1 \times NR_1} - 1\right)$.

5.2.2 Channel Model

In this section, we describe the QKD channel in the time-bin protocols. A closer inspection of errors suggests that the QKD channel is composed of a local channel and a global channel, that are governed by different error sources, referred to as the **local errors** and the **global errors**, respectively. Local errors originate from timing jitters and synchronization errors in entangled photon detection. Under local errors, the two photons from the same pair fall into different but close enough bins within the same frame. The offset delay between photons can be well-fitted by a Gaussian distribution, as shown in Fig. 5.4. Global errors are caused by accidental concurrent detections of stray photons not associated with any entangled photon pairs. The intra-frame time offset between the photons caused by global errors is close to a uniform distribution on $[0, T]$, where T refers to the frame size. Experimental results show a nontrivial dependency between the bin widths/frame sizes vs. the fraction of global errors

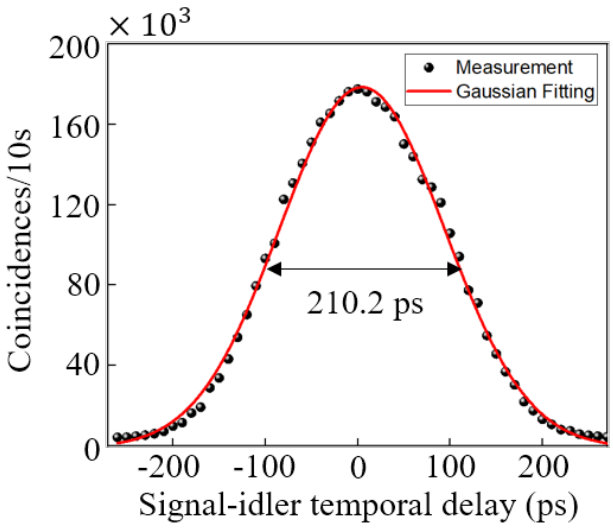


Figure 5.4: Distribution of the time offset caused by jitter errors.

out of the total number of selected frames.

Based on the aforementioned discussion, the QKD channel can be approximated by the transition probabilities $\mathbb{P}(Y = y|X = x)$, $x, y \in \mathrm{GF}(2^M)$ in (5.6). The channel is approximated by the mixture of a uniform distribution and a mixture of two Gaussian distributions,

$$\mathbb{P}(Y = y|Y = x) = \alpha_1 e^{-\frac{|y-x|^2}{\sigma_1^2}} + \alpha_2 e^{-\frac{|y-x|^2}{\sigma_2^2}} + \mu. \tag{5.6}$$

The quantities α_1, α_2 ,and μ are parameters determining the strength of the local Gaussian channel, the global Gaussian component, and the uniform component of the global channel, respectively. The only difference between this model and the local-global channel described in [5] is that the global channel in the current model also includes a Gaussian component instead of being a purely uniform channel as in [5]. The Gaussian-like distribution in the global channel is observed from the raw data obtained from our testbed and becomes more evident when the frame size increases. Moreover, in the raw data obtained from our testbed, we observe that the channel induces a very high raw symbol error rate, and the raw symbol error rate keeps increasing when M increases [95, 94, 102, 103]. We also observe a notable asymmetry between $\mathbb{P}(Y = x+i|Y = x)$ and $\mathbb{P}(Y = x-i|Y = x)$, $1 \leq i \leq \min\{x, 2^M-x-1\}$.

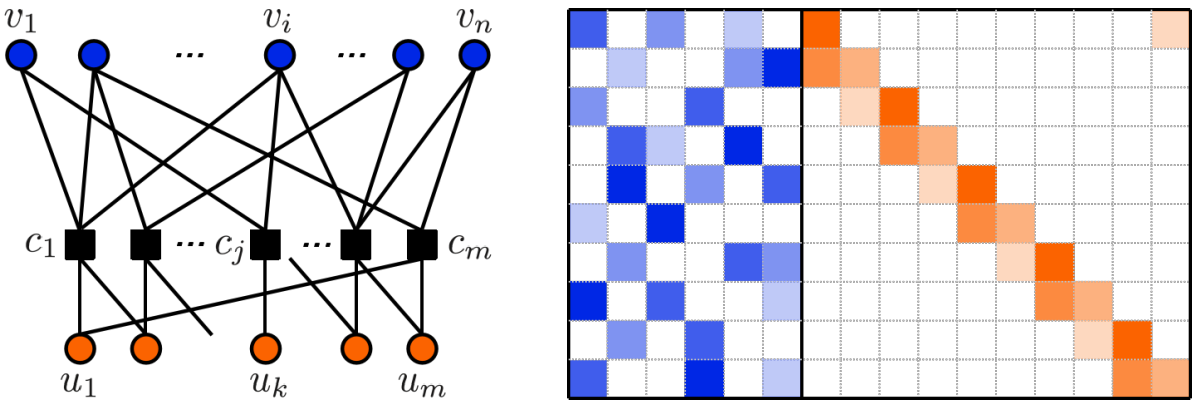


Figure 5.5: Tanner graph (left panel) and parity check matrix (right panel) of a IRA code.

This asymmetry is resultant from a misalignment of the center of the bins with the real arrival time of photons generated at Alice's side.

5.3 SC-Irregular-Repeat-Accumulate (SC-IRA) Codes

In this section, we present the SC-IRA codes proposed in this work. In this work, we aim at minimizing the number of short cycles in the unlabelled graph, and the non-zero non-binary entries are randomly assigned. LDPC codes designed for the waterfall region (with the error rate of > 0.001) are typically optimized through degree distribution [104, 105], while finite-length optimization typically only improves the low FER region (referred to as the error-floor region) [3]. While the GRADE-AO methods were observed to have a universal gain over the whole range of FER for high rate codes used in data storage and memories, their potential in optimizing low rate codes (for high FER region) has not been discussed. In this work, we generalize the framework for arbitrary underlying block codes. In particular, we propose to use IRA codes as the underlying block codes to harness the full power of SC-IRA codes in QKD applications.

In the low FER region, the coexistence of CNs of low degrees and VNs of moderate/high degrees are critical for successful decoding, which requires an adequate placement of a large number of degree 2 VNs [105]. Performance of randomly constructed LDPC codes can be far

away from the theoretical limit because of a casual placement of degree 2 VNs. IRA codes, in which most of the degree 2 VNs are placed on a long ring, are known to be capacity-achieving for various channels [106, 107, 108], and are ideal for QKD channels, as we show next.

$$\mathbf{\Pi} = \left[\begin{array}{cccccc|cccccccccc} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{array}\right]. \tag{5.7}$$

As shown in Fig. 5.5, an IRA code is composed of two parts: the irregular part consisting of VNs $v_1, v_2, \ldots, v_n$, all the CNs and the edges connecting them, the other part being a large ring of length 2γ containing VNs $u_1, u_2, \ldots, u_m$ and all the CNs. The parity check matrix $\mathbf{\Pi}$ in (5.7) specifies an IRA code with $m = 6$, $n = 10$. This systematic placement of degree 2 VNs exploits the benefits of having such nodes, and enables them to be much more reliable than degree 2 VNs in randomly constructed LDPC codes.

Fig. 5.6 shows an SC-IRA code with the underlying block code specified in (5.7) and $(\gamma, \kappa, m, L) = (10, 16, 3, 5)$. The unlabelled graph of a non-binary (NB) SC code can be represented by its partitioning matrix $\mathbf{P} \in \{*, 0, 1, \ldots, m\}^{\gamma \times \kappa}$, where $\mathbf{P}(i, j) = k$ iff. $\mathbf{\Pi}_k(i, j) = 1$, and $\mathbf{P}(i, j) = *$ iff. $\mathbf{\Pi}(i, j) = 0$. The partitioning matrix of the SC code in FIG. 5.6 is

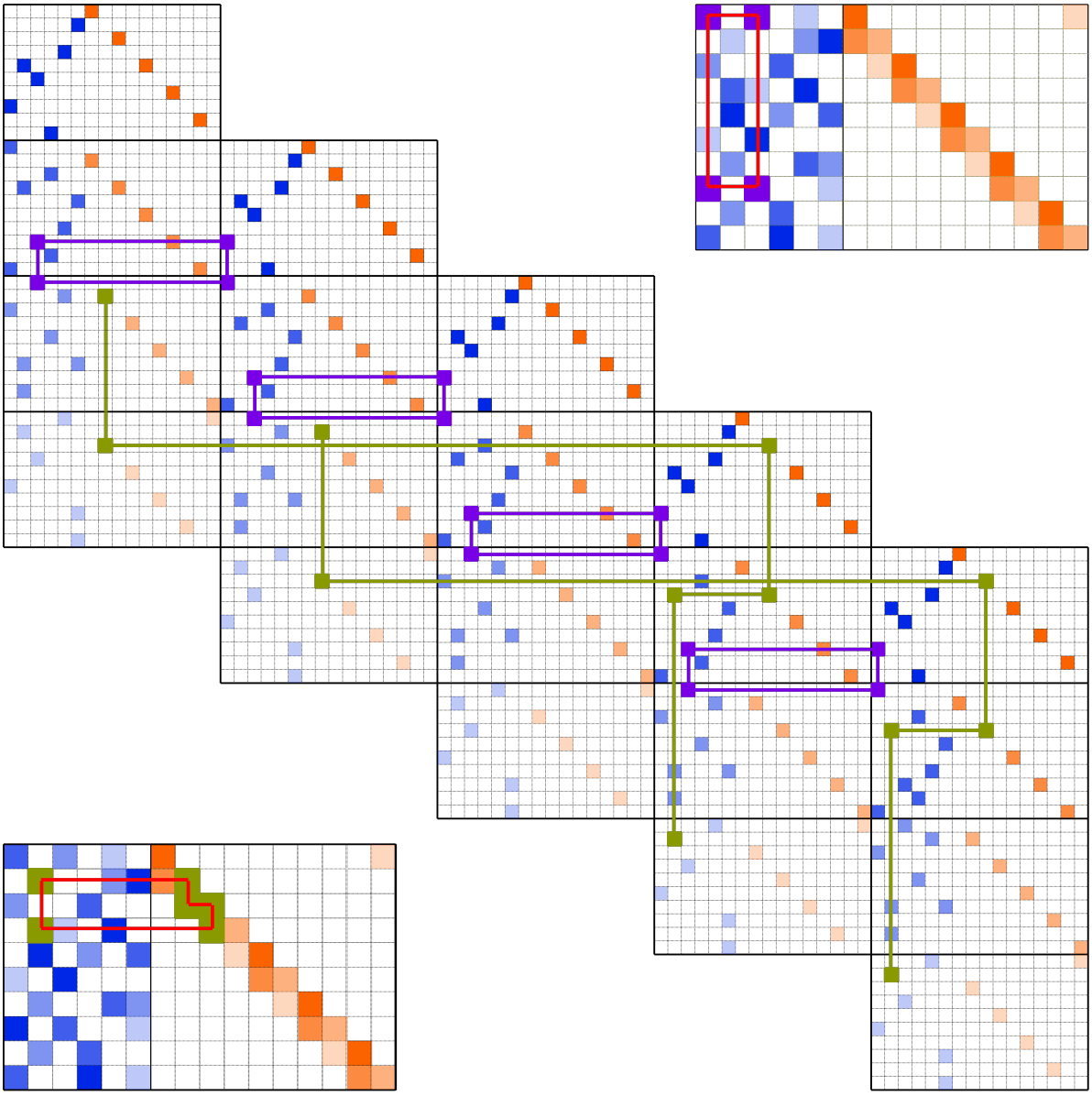


Figure 5.6: A memory 3 SC-IRA code with the underlying IRA code specified as in FIG. 5.5. The cycle-4 highlighted in the base matrix on the top-right panel results in cycles-4 in the SC-IRA codes, whereas the cycle-6 in the base matrix on the bottom-left panel does not lead to any cycles in the SC-IRA code.

specified in (5.8),

$$\mathbf{P} = \left[\begin{array}{cccccc|cccccccccc} 1 & * & 2 & * & 3 & * & 0 & * & * & * & * & * & * & * & * & 3 \\ * & 3 & * & * & 2 & 0 & 1 & 2 & * & * & * & * & * & * & * & * \\ 2 & * & * & 1 & * & * & * & 3 & 0 & * & * & * & * & * & * & * \\ * & 1 & 3 & * & 0 & * & * & * & 1 & 2 & * & * & * & * & * & * \\ * & 0 & * & 2 & * & 1 & * & * & * & 3 & 0 & * & * & * & * & * \\ 3 & * & 0 & * & * & * & * & * & * & * & 1 & 2 & * & * & * & * \\ * & 2 & * & * & 1 & 2 & * & * & * & * & * & 3 & 0 & * & * & * \\ 0 & * & 1 & * & * & 3 & * & * & * & * & * & * & 1 & 2 & * & * \\ * & 2 & * & 1 & * & * & * & * & * & * & * & * & * & 3 & 0 & * \\ 1 & * & * & 0 & * & 3 & * & * & * & * & * & * & * & * & 1 & 2 \end{array}\right]. \tag{5.8}$$

GRADE-AO is a joint probabilistic-combinatorial optimization framework that minimizes the number of detrimental objects, i.e., the dominating sources of decoding failures, in SC codes with moderate/high memories. Unlike previous methods that primarily focus on the degree distribution, GRADE-AO considers a code design parameter that is more relevant for the design of SC codes. In particular, GRADE-AO effectively exploits the impact of edge density distribution (among the component matrices) on the performance of SC codes. GRADE-AO framework explores an edge density distribution that locally minimizes the expected number of targeted short cycles in the resultant SC codes, followed by an algorithmic optimization that further reduces the number of detrimental objects without a dramatic change in the targeted distribution. Monte-Carlo simulations show a significant performance gain of SC codes with the skewed edge distribution obtained from GRADE over codes with the uniform edge distribution. Asymptotic analysis also demonstrated benefits of nonuniform edge distribution [75]. We next introduce the details of the GRADE-AO scheme for SC-IRA codes.

We refer to a cycle of length $2g$ in the Tanner graph of the code as a cycle-$2g$. A cycle-$2g$

in an SC code is associated with a closed path of length $2g$ on the nonzero entries in its partitioning matrix, referred to as a closed length-$2g$ path. A closed length-$2g$ path, $g \in \mathbb{N}$, $g \geq 2$ is denoted by $C(j_1, i_1, j_2, i_2, \ldots, j_g, i_g)$, where (i_k, j_k), (i_k, j_{k+1}), $1 \leq k \leq g$, $j_{g+1} = j_1$. A closed length-$2g$ path leads to cycles-$2g$ in the SC code iff. the following **cycle condition** follows [45]:

$$\sum_{k=1}^{g} \mathbf{P}(i_k, j_k) = \sum_{k=1}^{g} \mathbf{P}(i_k, j_{k+1}). \tag{5.9}$$

Definition 15. *Assign a weight $w(d)$ to each degree d VN, and let the weight of a cycle be the summation of the weights of all the VNs on it; $w : \mathbb{N} \to \mathbb{R}$ is a strictly decreasing function, e.g., $w(d) = 1/d$. Let $\Lambda \in \mathbb{R}[x]$, where Λ_i denotes the fraction of degree i VNs out of all the VNs in an irregular base matrix $\mathbf{\Pi}$. Let $m \in \mathbb{N}$. Let $\mathbf{p}_d = (p(0|d), p(1|d) \ldots, p(m|d))$, where $d \in \mathbf{N}$, $d \leq \deg \Lambda$, $0 < p(i|d) \leq 1$, $p(0|d) + p(1|d) + c \cdots + p(m|d) = 1$: each $p(i|d)$ specifies the probability of a '1' in a column with weight d in $\mathbf{\Pi}$ going to the component matrix $\mathbf{\Pi}_i$; $\mathbf{p}$ is referred to as **edge distribution** under random partition later on. Then, the following $f(X, Y; \{\mathbf{p}_d\}_{d=1}^{\deg \Lambda}, \Lambda, w)$, which is abbreviated to $f(X, Y)$ when the context is clear, is called the **coupling polynomial** of an SC code with the coupling pattern $\mathbf{a}$, associated with probability distribution $\mathbf{p}$:*

$$f(X, Y; \mathbf{p}, \Lambda, w) = \sum_{d} \Lambda_d Y^{w(d)} \left(\sum_{i=0}^{m} p(i|d) X^i \right) \left(\sum_{i=0}^{m} p(i|d) X^{-i} \right). \tag{5.10}$$

Based on the new definition of the coupling polynomial, we derive the expected weight of a cycle-$2g$ in the base matrix when it is expanded to the Tanner graph. We denote this weight by $W_{2g}(\mathbf{p}; \mathbf{\Lambda}, w)$, $g \in \mathbb{N}$. We also specify the gradient of $W_{2g}(\mathbf{p}; \mathbf{\Lambda}, w)$. Given $\mathbf{\Lambda}$ and $w(d)$, a locally-optimal set of conditional edge distributions $\{p(i|d)\}_{i=0}^{m}$, $d \in \mathbb{N}$, can be derived by applying a gradient descent algorithm to minimize $W_{2g}(\mathbf{p}; \mathbf{\Lambda}, w)$.

Theorem 9. *For $g \in \mathbb{N}$, $g \geq 2$, denote by $W_{2g}(\mathbf{p}, \Lambda, w)$ the expected weight of a cycle-$2g$ in the base matrix when they are expanded to the Tanner graph under random partitioning with*

edge distribution $\mathbf{p}$*. Then,*

$$\begin{aligned} W_{2g}(\mathbf{p};\mathbf{\Lambda},w) &= \left[\partial_Y f^g(X,Y)|_{Y=1}\right]_{X^0}, \\ \nabla_{\mathbf{p}} W_{2g}(\mathbf{p};\mathbf{\Lambda},w) &= \left[\partial_Y \nabla_{\mathbf{p}} f^g(X,Y)|_{Y=1}\right]_{X^0}, \end{aligned} \tag{5.11}$$

where $[g(X)]_{X^0}$ *denotes the constant term of* $g(X)$*.*

Proof. While the formula of the gradient $\nabla_{\mathbf{p}} W_{2g}(\mathbf{p};\mathbf{\Lambda},w)$ is obviously true, we only need to prove the formula obtaining the expected weight of a cycle-$2g$. Suppose a cycle-$2g$ in the base matrix is represented by $C(j_1,i_1,j_2,i_2,\ldots,j_{2g},i_{2g})$. Let $\mathbb{I}(\cdot)$ denotes the indicator function, i.e., $\mathbb{I}(S)=1$ iff. the expression S is true, otherwise $\mathbb{I}(S)=0$. Then, the weight $W(C)$ of the cycle $C(j_1,i_1,j_2,i_2,\ldots,j_{2g},i_{2g})$ in the Tanner graph can be represented by the following function:

$$W(C) = \left(\sum_{k=1}^{g} w(d_{j_k})\right) \mathbb{I}\left(\textstyle\sum_{k=1}^{g} \mathbf{P}(i_k,j_k) = \sum_{k=1}^{g} \mathbf{P}(i_{k+1},j_k)\right), \tag{5.12}$$

where d_{j_k} denotes the node degree of the VN corresponding to the j_k'th column in the base matrix. Then, the expected weight $W_{2g}(\mathbf{p};\mathbf{\Lambda},w)$ for each cycle is the expectation value of $W(C)$ over all possible realizations of $\mathbf{P}$ and the set of VN degrees $\{d_{j_k}\}_{k=1}^{g}$. That is to say,

$$\begin{aligned} & W_{2g}(\mathbf{p};\mathbf{\Lambda},w) \\ =& \mathbb{E}_{\{d_{j_k}\}_{k=1}^{g},\mathbf{P}}\left[W(C)\right] \\ =& \mathbb{E}_{\{d_{j_k}\}_{k=1}^{g},\mathbf{P}}\left[\left(\sum_{k=1}^{g} w(d_{j_k})\right) \mathbb{I}\left(\textstyle\sum_{k=1}^{g} \mathbf{P}(i_k,j_k) = \sum_{k=1}^{g} \mathbf{P}(i_{k+1},j_k)\right)\right] \\ =& \mathbb{E}_{\{d_{j_k}\}_{k=1}^{g}}\left[\left(\sum_{k=1}^{g} w(d_{j_k})\right) \mathbb{P}\left[\textstyle\sum_{k=1}^{g} \mathbf{P}(i_k,j_k) = \sum_{k=1}^{g} \mathbf{P}(i_{k+1},j_k)\right]\right] \\ =& \mathbb{E}_{\{d_{j_k}\}_{k=1}^{g}}\left[\left(\sum_{k=1}^{g} w(d_{j_k})\right) \sum_{\sum_{k=1}^{g} x_k = \sum_{k=1}^{g} y_k} \prod_{k=1}^{g} \mathbb{P}\left[\mathbf{P}(i_k,j_k)=x_k, \mathbf{P}(i_{k+1},j_k)=y_k\right]\right] \end{aligned}$$

$$
\begin{aligned}
&= \sum_{\{d_k\}_{k=1}^g} \left(\sum_{k=1}^g w(d_k) \right) \sum_{\sum_{k=1}^g x_k = \sum_{k=1}^g y_k} \\
&\quad \prod_{k=1}^g \mathbb{P}\left[\mathbf{P}(i_k, j_k) = x_k, \mathbf{P}(i_{k+1}, j_k) = y_k | d_{j_k} = d_k\right] \mathbb{P}\left[d_{j_k} = d_k\right] \\
&= \sum_{\{d_k\}_{k=1}^g, \sum_{k=1}^g x_k = \sum_{k=1}^g y_k} \left(\sum_{k=1}^g w(d_k) \right) \prod_{k=1}^g \left[p(x_k|d_k) p(y_k|d_k) \Lambda_{d_k} \right] \\
&= \left[\sum_{\{d_k\}_{k=1}^g, 0 \le x_k, y_k \le m} \left(\sum_{k=1}^g w(d_k) \right) \prod_{k=1}^g \left[p(x_k|d_k) p(y_k|d_k) \Lambda_{d_k} \right] X^{\sum_{k=1}^g (x_k - y_k)} \right]_0 \\
&= \left[\partial_Y \left\{ \sum_{\{d_k\}_{k=1}^g, 0 \le x_k, y_k \le m} \prod_{k=1}^g \left[p(x_k|d_k) p(y_k|d_k) \Lambda_{d_k} \right] X^{\sum_{k=1}^g (x_k - y_k)} Y^{\sum_{k=1}^g w(d_k)} \right\} |_{Y=1} \right]_0 \\
&= \left[\partial_Y \left\{ \sum_{\{d_k\}_{k=1}^g, 0 \le x_k, y_k \le m} \prod_{k=1}^g \left[p(x_k|d_k) X^{x_k} p(y_k|d_k) X^{-y_k} \Lambda_{d_k} Y^{w(d_k)} \right] \right\} |_{Y=1} \right]_0 \\
&= \left[\partial_Y f^g(X, Y) |_{Y=1} \right]_{X^0} .
\end{aligned}
$$

The theorem is proved. □

Remark 15. *Note that the nonzero elements of* $\mathrm{GF}(2^M)$ *form a cyclic group of order* $(2^M - 1)$*, and each of them can be represented by a unique power* a^i*,* $i \in \{1, 2, \ldots, 2^M - 1\}$*, of a primitive element* $a \in \mathrm{GF}(2^M)^\times$*. Suppose an active cycle-*$2g$ *in the unlabelled Tanner graph is assigned with edge weights* $a^{i_1}, a^{i_2}, \ldots, a^{i_{2g}}$*. Then, the unlabelled cycle becomes a cycle in the NB code iff. (5.13) is satisfied.*

$$
\sum_{k=1}^g i_{2k-1} \equiv \sum_{k=1}^g i_{2k} \mod (2^M - 1). \tag{5.13}
$$

Then, the edge weights can be jointly optimized with the partitioning matrix in the AO step.

5.4 Experimental Results

In this section, we present the simulation results that demonstrate the superiority of our proposed codes over the existing works. We compare our proposed NB-SC-IRA codes with the MLC codes of length 2000 and 2500. Fig. 5.7 and Fig. 5.8 present the best normalized

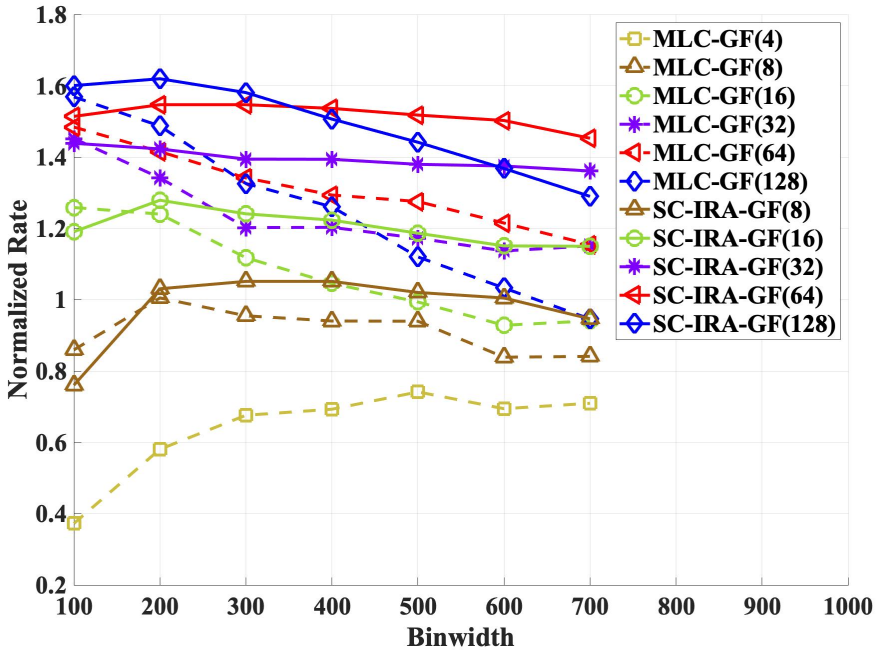


Figure 5.7: Comparison of normalized rates at blocklength 2000.

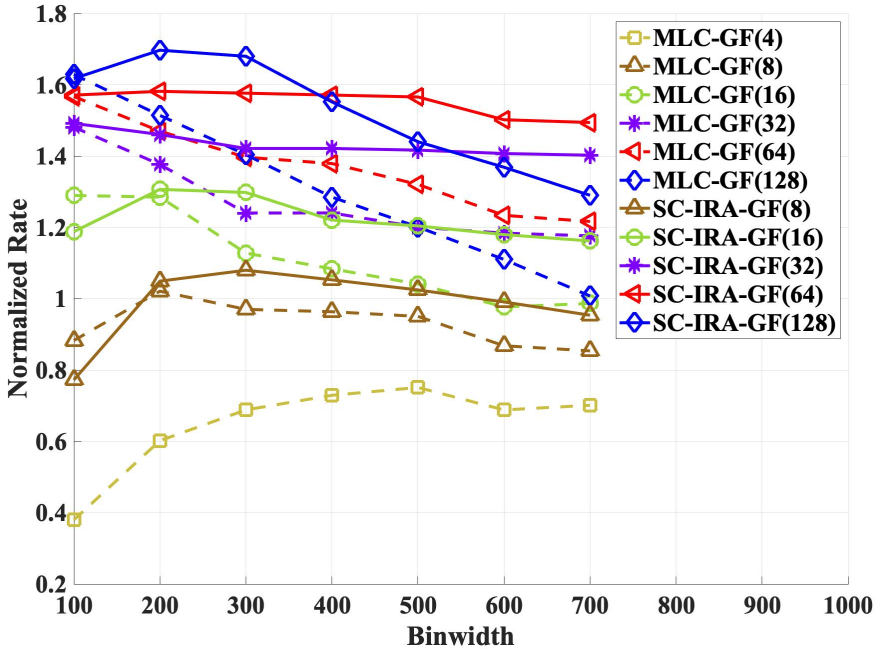


Figure 5.8: Comparison of normalized rates at blocklength 2500.

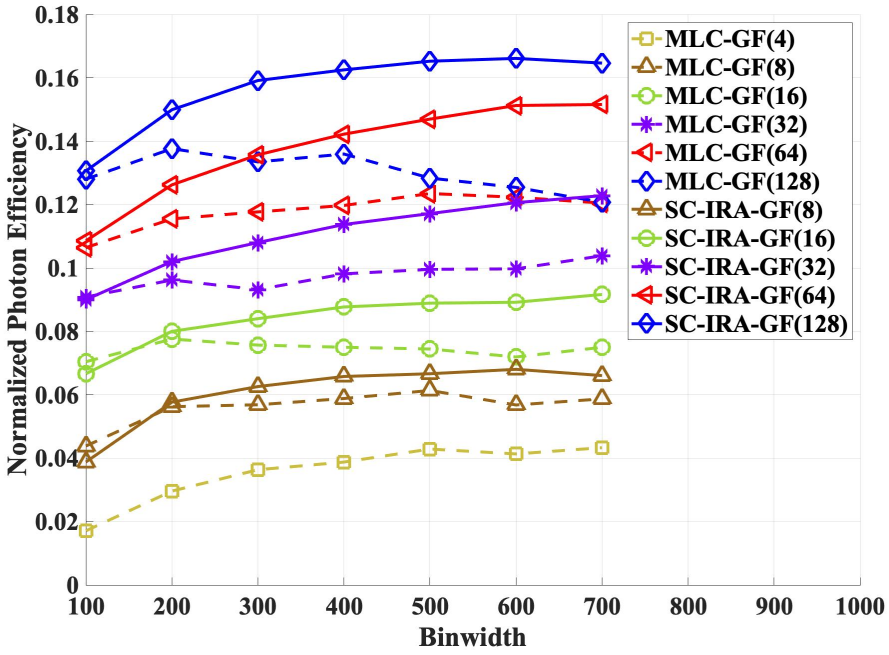


Figure 5.9: Comparison of normalized photon efficiencies at blocklength 2000.

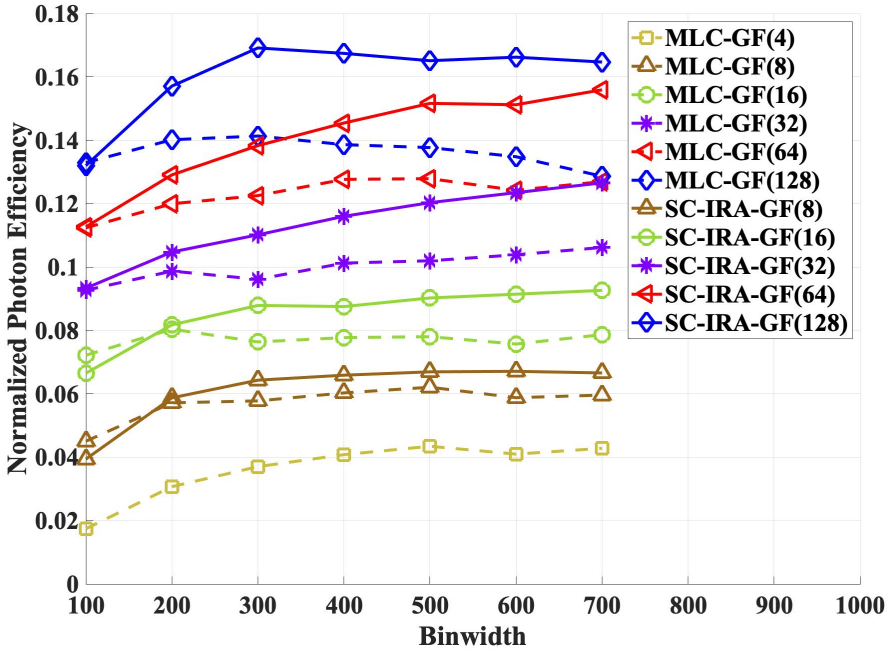


Figure 5.10: Comparison of normalized photon efficiencies at blocklength 2500.

rates achieved by our SC-IRA scheme and the MLC scheme at blocksize near 2000 and 2500, bin widths from $\{100, 200, \ldots, 700\}$, and bit number $M \in \{2, 3, \ldots, 7\}$, respectively. We observe a significant gain in the normalized rates of SC-IRA codes over that of MLC codes, especially when the bin width is greater than 100ps. The normalized rates of SC-IRA codes are shown to be much more stable against the change of the bin widths, i.e., they are more robust against global errors. This is a desirable outcome since the RPE increases when the bin width increases from 100ps to 700ps. Fig. 5.9 and Fig. 5.10 present the normalized photon efficiencies corresponding to the SC-IRA scheme and the MLC scheme shown in Fig. 5.7 and Fig. 5.8, respectively. We observe a significant gain of 19.67% and 20.62% of SC-IRA codes over that of MLC codes, in NPE. Note that as discussed previously, the gain in SKR is strictly larger than NPE, which further improves our gain in the low rate region.

The overall experimental setup is shown in Fig. **8**. We use a fiber-coupled, periodically-poled $KTiOPO_4$ (ppKTP) waveguide for type-II spontaneous parametric downconversion (SPDC) generation [109, 94, 95]. The waveguide is pumped by a 658.288nm, $\sim$ 6.3mW Fabry-Perot laser diode stabilized by self-injection locking [94, 95]. As a result, the source can provide up to $100k\ pairs/s$ with 5% coincidences to singles ratio. The type-II generated SPDC pairs are orthogonally polarized frequency-degenerate signal and idler photons that are centered around 1316.628nm over a 245GHz ($\sim$ 1.4nm) full-width-to-half-maximum (FWHM) phase-matching bandwidth [110, 95]. Bandpass filters (BPF) with 1.3nm FWHM linewidth are used to filter residual pump photons and ppKTP out-of-band fluorescence. The ppKTP source and BPF are synchronized and stabilized, benefiting from second-harmonic generation to coincide the central bin to the SPDC spectral maximum. Orthogonally-polarized signal and idler photons are separated using a cube polarizing beamsplitter and sent to Alice and Bob's channels. The time synchronization between Alice and Bob is achieved using temporal correlation measurements between signal and idler photons. In order to read the information, superconducting nanowire single-photon detectors (SNSPD) with 85% detection efficiency and 70–80ps timing jitter are used. The measured uncertainty of coincidence between signal

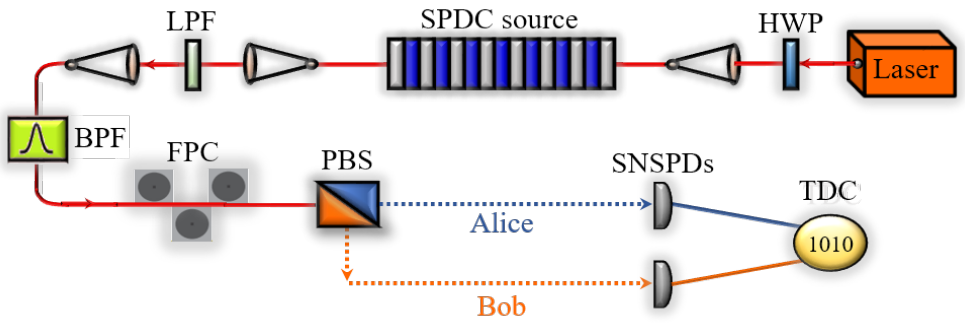


Figure 5.11: The time-bin encoding testbed for SC-IRA codes. The pump laser is stabilized and laser polarization is optimized using half-wave plate (HWP). The pump laser passed through spontaneous parametric down-conversion (SPDC) source for biphoton generation, while the residual pump laser is filtered using long-pass filter (LPF). The biphoton stream is further isolated using a bandpass filter (BPF) and its polarization axis is optimized against drifts using a fiber polarization controller (FPC). The orthogonally polarized signal and idler photons are separated using fiber polarizing beamsplitter(FPBS). The distributed signal and idler photons are detected by Alice and Bob using superconducting nanowire single-photon detectors (SNSPD) and converted to symbols using time-to-digital converter (TDC).

and idler photons is 210.2ps [102, 103].

We construct the underlying IRA codes with blocksizes $\kappa = k + i$, $k \in \{40, 50\}$, $i \in \{0, 1, \ldots, 4\}$, and the corresponding SC-IRA codes with $m = 5$, $L = 50 + k - \kappa$: this set-up allows us to have a pool of SC-IRA codes with sufficiently small stepsizes in rates, from which we select the candidate that reaches the best normalized rate. The SC-IRA codes are of lengths $N \in \{(k+i)(L-i) = KL + (L-k-i)i | i = 0, 1, \ldots, 4\}$, i.e., $N \in \{2000, 2009, 2016, 2024, 2025\}$ and $N \in \{2500, 2499, 2496, 2491, 2484, 2475\}$, respectively. The SC-IRA codes are optimized with respect to the number of short cycles; most of the code candidates are free of cycles up to length 8, and the number of cycles-10 and cycles-12 are also jointly minimized, resulting in SC-IRA codes with large minimum distances. The irregular part of the underlying IRA codes are randomly constructed near-regular LDPC codes with VNs of degrees 4 and 5, and CNs of degrees 2 and 3, resulting in an average VN degree of around 4.2. We use a near-regular IRA code as the underlying block code in our simulations for simplicity and fair comparison; the performance can be further improved by optimizing the degree distribution of the underlying IRA codes for the QKD

channel. The codes are decoded by floating-point SPA, in which Fast Fourier Transform (FFT) over GF(2^M) has been applied in updating the CN messages to improve the decoding complexity in large alphabets [85].

Note that existing protocols using SC codes [54, 56] typically overlooked the finite-length optimization since the degree distribution has been considered as the major parameter that determines the performance in waterfall region. In this work, we exploit another degree of freedom appeared in SC codes, the edge density distribution, that also significantly affects the waterfall performance. Despite we have not optimized the degree distribution of the underlying IRA codes in this paper, our framework can be applied in conjunction with degree distribution optimization to further improve the SKR.

5.5 Conclusion

In this work, we study ECC solutions that are appropriate for energy-time entanglement based photonic QKD protocols. In particular, we have taken into account the effects of modulation schemes on the QKD channels into consideration while designing ECC codes, which have been overlooked in existing literature. In order to achieve high secure key rate (SKR), which is critical in photon-starved scenarios, modulation schemes that result in a larger number of selected photon pairs and of higher dimension are desired. However, these modulation schemes lead to a larger fraction of global errors, which are nearly uniformly distributed over the alphabet. While existing solutions stemmed from the MLC scheme have done a great job in combating local errors caused by timing jitters, they are unable to sustain a high reconciliation efficiency when combined with modulation schemes with higher raw key rates, due to being highly reliant on the dependencies between bit layers. In the contrary, we propose a class of non-binary LDPC codes, the SC-IRA codes, which is more robust than the MLC scheme in combating uniform errors, and leads to higher SKR.

Our SC-IRA codes are with high memories and have been carefully optimized through the GRADE-AO framework proposed in [57] with respect to the number of short cycles up

to length 12 in their Tanner graphs. GRADE-AO is an efficient optimization framework that constructs high performance SC codes with high memories. When comparing our codes with the MLC scheme in the short codelength regime, which is desired for low latency decoding and is especially important in quantum networks, it is apparent that the reconciliation efficiencies of the SC-IRA codes deteriorate much slower compared with the MLC schemes with respect to the bin widths. We adopt a measure NPE to estimate the lower bound on the gain between the SKR of protocols with identical photon generation rate. The improvement in the raw key rate, accompanied by the gain in the reconciliation efficiency, leads to a significant 20% gain in the NPE over the MLC scheme. Our work applies to the scenarios where accidental photon pairs are desired to increase the raw key rate, which is especially true in high-dimensional cases. Future work includes combining our work with the adaptive modulation scheme proposed in [100] and degree distribution optimization to further improve the SKR.

CHAPTER 6

Conclusion

6.1 Summary of Contributions

The primary goal of this dissertation was to devise advanced error correction coding solutions for modern data storage and communication systems, in order to reach desirable properties like high reliability, low latency and scalability. In particular, the dissertation consists of two major coding schemes for three different practical data-driven applications. The first problem focused on hierarchical codes, a class of locally recoverable codes that are appropriate for cloud storage due to their ability of achieving low latency in reading data. We developed hierarchical codes for both centralized and decentralized cloud storage. The second problem was about overcoming the computational challenge in optimizing SC-LDPC codes with high memories. Built upon our proposed framework, we constructed state-of-the-art SC codes tailored for memory devices and quantum key distributions, respectively.

In the line of hierarchical coding for modern cloud storage, we developed a new class of hierarchical codes based on Cauchy matrices. Our proposed codes were able to seamlessly adapt to dynamic changes of usage rates in different components in heterogeneous cloud storage. In addition, the Galois field size of this scheme grows linearly with the maximum local block length. We proposed a double-level construction first, followed by a triple-level construction based on it; this construction can be generalized into any hierarchical structure with a larger number of layers.

While our proposed codes have been proved efficient in centralized cloud storage, we then moved on to decentralized storage networks (DSNs), inspired by the prevalence of decentralized systems under the recent rise of the blockchain technology. We proposed a joint coding scheme that enables neighboring nodes to cooperatively protect and validate their stored data, which improves the reliability collectively due to propagation of information from more reliable nodes to less reliable ones. While our proposed constructions obviously preserve scalability and flexibility that are critical in dynamic networks, they also adapt to arbitrary topologies, a property that is especially important in DSNs but has been overlooked in existing works.

In the other line of SC-LPDC codes, we overcame the computational challenges in discrete optimization on finite-length high-performance SC codes with high memories. We investigated the relation between the edge distribution and the expected number of targeted objects in the SC ensembles, based on which we obtained a locally optimal edge distribution by gradient descent. The edge distribution identified a search subspace that is much smaller than the original space, thus reduces the complexity of the algorithmic optimizer in the next step significantly without unbearable loss in performance. Our framework, referred to as GRADE-AO, is generally applicable to a large variety of targeted objects. In particular, we applied GRADE-AO to optimize the number of pairs of concatenated cycles, by which we managed to construct codes with excellent performance on NLM channels and MR channels.

With the success of GRADE-AO in constructing high rate codes that are appropriate for memory devices and hard disk drives, we built upon it low rate SC codes in combination with IRA codes such that they can be applied to QKD. In particular, We extended our GRADE-AO framework to be applicable in irregular underlying codes, by which we developed SC-IRA codes with moderate lengths and large girth. Our proposed codes were demonstrated to outperform MLC/MSD schemes by experiment results.

6.2 Future Directions

Regarding hierarchical coding for cloud storage, a major bottleneck of our codes is that for each targeted node, at most one out of its neighboring nodes is allowed to have a number erasures that exceeds the associated local erasure-correction capability. Devising component codes where multiple neighboring nodes are allowed to be non-locally recoverable simultaneously is therefore one of the most important future directions.

For the SC codes for data storage, while we have observed the nontrivial performance gain of codes with nonuniform edge distribution over their uniform counterparts in the waterfall region, there still lacks a theoretical explanation to this phenomenon. Therefore, the next step will be the asymptotic analysis on SC ensembles with edge distribution being taken into consideration. As discussed previously, global algorithmic optimizers of TC codes and the associated low latency decoders are also of interests.

For SC codes tailored for QKD, one major appealing direction is the combination of SC-IRA codes with the adaptive modulation scheme that is aimed at increasing the raw key rate. In particular, the adaptive modulation scheme utilizes the multi-detection frames that are discarded in traditional protocols, by strategically changing the frame size. This adaptation requires manipulating input symbols from different alphabet sizes, in which a hybrid coding solution will be appropriate. Having said that, symbols generated from photon pairs in frames with different sizes visualize different channels, thus naturally calls for a solution that adapts to channel variation accordingly.

References

[1] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Trans. Inf. Theory*, vol. 56, no. 9, pp. 4539–4551, Sept 2010.

[2] I. Tamo and A. Barg, "A family of optimal locally recoverable codes," *IEEE Trans. Inf. Theory*, vol. 60, no. 8, pp. 4661–4676, Aug. 2014.

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

[4] M. Lentmaier, A. Sridharan, D. J. Costello, and K. S. Zigangirov, "Iterative decoding threshold analysis for LDPC convolutional codes," *IEEE Trans. Information Theory*, vol. 56, no. 10, pp. 5274–5289, Oct. 2010.

[5] H. Zhou, L. Wang, and G. Wornell, "Layered schemes for large-alphabet secret key distribution," in *2013 Information Theory and Applications Workshop (ITA)*. IEEE, 2013, pp. 1–10.

[6] R. Gabrys, E. Yaakobi, and O. Milenkovic, "Codes in the Damerau distance for deletion and adjacent transposition correction," *IEEE Trans. Inf. Theory*, vol. 64, no. 4, pp. 2550–2570, April 2018.

[7] S. Yang, C. Schoeny, and L. Dolecek, "Theoretical bounds and constructions of codes in the generalized cayley metric," *IEEE Transactions on Information Theory*, vol. 65, no. 8, pp. 4746–4763, Aug. 2019.

[8] ——, "Order-optimal permutation codes in the generalized cayley metric," in *2017 IEEE Information Theory Workshop (ITW)*, Nov. 2017, pp. 234–238.

[9] M. Hassner, K. Abdel-Ghaffar, A. Patel, R. Koetter, and B. Trager, "Integrated interleaving-a novel ECC architecture," *IEEE Transactions on Magnetics*, vol. 37, no. 2, pp. 773–775, 2001.

[10] P. Huang, E. Yaakobi, and P. H. Siegel, "Multi-erasure locally recoverable codes over small fields," in *2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2017, pp. 1123–1130.

[11] Y. Cassuto, E. Hemo, S. Puchinger, and M. Bossert, "Multi-block interleaved codes for local and global read access," in *Proc. IEEE Int. Symp. Inf. Theory*, 2017, pp. 1758–1762.

[12] Y. Wu, "Generalized integrated interleaved codes," *IEEE Transactions on Information Theory*, vol. 63, no. 2, pp. 1102–1119, Nov. 2017.

[13] S. Ballentine, A. Barg, and S. Vladuts, "Codes with hierarchical locality from covering maps of curves," *arXiv preprint arXiv:1807.05473*, 2018.

[14] X. Zhang, "Generalized three-layer integrated interleaved codes," *IEEE Communications Letters*, vol. 22, no. 3, pp. 442–445, 2018.

[15] M. Blaum and S. R. Hetzler, "Extended product and integrated interleaved codes," *IEEE Trans. Inf. Theory*, vol. 64, no. 3, pp. 1497–1513, Mar. 2018.

[16] S. B. Balaji, G. R. Kini, and P. V. Kumar, "A tight rate bound and matching construction for locally recoverable codes with sequential recovery from any number of multiple

erasures," *IEEE Transactions on Information Theory*, vol. 66, no. 2, pp. 1023–1052, 2020.

[17] B. P. Rimal, E. Choi, and I. Lumb, "A taxonomy and survey of cloud computing systems," in *INC, IMS and IDC, 2009. NCM'09. Fifth International Joint Conference on*, 2009, pp. 44–51.

[18] "Storj: A decentralized cloud storage network framework," Oct. 2018. [Online]. Available: https://storj.io/storjv3.pdf

[19] H. Shafagh, L. Burkhalter, A. Hithnawi, and S. Duquennoy, "Towards blockchain-based auditable storage and sharing of IoT data," in *Proceedings of the 2017 on Cloud Computing Security Workshop*, Nov., pp. 45–50.

[20] G. Zyskind, O. Nathan *et al.*, "Decentralizing privacy: Using blockchain to protect personal data," in *IEEE Security and Privacy Workshops*, May 2015, pp. 180–184.

[21] Y. Zhu, C. Lv, Z. Zeng, J. Wang, and B. Pei, "Blockchain-based decentralized storage scheme," in *Journal of Physics: Conference Series*, vol. 1237, no. 4, Apr. 2019, p. 042008.

[22] U. Martnez-Penas and F. R. Kschischang, "Universal and dynamic locally repairable codes with maximal recoverability via sum-rank codes," in *2018 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2018, pp. 792–799.

[23] J. Pernas, C. Yuen, B. Gastón, and J. Pujol, "Non-homogeneous two-rack model for distributed storage systems," in *IEEE International Symposium on Information Theory*, Jun. 2013, pp. 1237–1241.

[24] Y. Wang, D. Wei, X. Yin, and X. Wang, "Heterogeneity-aware data regeneration in distributed storage systems," in *Proceedings of 2010 IEEE INFOCOM*, 2014, pp. 1878–1886.

[25] A. M. Ibrahim, A. A. Zewail, and A. Yener, "Green distributed storage using energy harvesting nodes," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 5, pp. 1590–1603, May 2016.

[26] M. Sipos, J. Gahm, N. Venkat, and D. Oran, "Network-aware feasible repairs for erasure-coded storage," *IEEE/ACM Transactions on Networking*, vol. 26, no. 3, pp. 1404–1417, Mar. 2018.

[27] ——, "Erasure coded storage on a changing network: The untold story," in *IEEE Global Communications Conference (GLOBECOM)*, Dec. 2016, pp. 1–6.

[28] J. Li, S. Yang, X. Wang, and B. Li, "Tree-structured data regeneration in distributed storage systems with regenerating codes," in *Proceedings of 2010 IEEE INFOCOM*, 2010, pp. 1–9.

[29] A. Tebbi, T. H. Chan, and C. W. Sung, "Multi-rack distributed data storage networks," *IEEE Transactions on Information Theory*, vol. 65, no. 10, pp. 6072–6088, Oct. 2019.

[30] H. Hou, P. P. C. Lee, K. W. Shum, and Y. Hu, "Rack-aware regenerating codes for data centers," *IEEE Transactions on Information Theory*, vol. 65, no. 8, pp. 4730–4745, Aug. 2019.

[31] Z. Chen and A. Barg, "Explicit constructions of MSR codes for clustered distributed storage: The rack-aware storage model," 2019. [Online]. Available: https://arxiv.org/abs/1901.04419

[32] N. Prakash, V. Abdrashitov, and M. Médard, "The storage versus repair-bandwidth trade-off for clustered storage systems," *IEEE Transactions on Information Theory*, vol. 64, no. 8, pp. 5783–5805, Aug. 2018.

[33] S. Yang, A. Hareedy, R. Calderbank, and L. Dolecek, "Hierarchical coding to enable scalability and flexibility in heterogeneous cloud storage," in *IEEE Global Communications Conference (GLOBECOM)*, Dec. 2019.

[34] ——, "Topology-aware cooperative data protection in blockchain-based decentralized storage networks," in *2020 IEEE International Symposium on Information Theory (ISIT)*, Jul. 2020, pp. 622–627.

[35] ——, "Hierarchical coding for cloud storage: Topology-adaptivity, scalability, and flexibility," *arXiv preprint arXiv:2009.09146*, 2020.

[36] F. T. Hady, A. Foong, B. Veal, and D. Williams, "Platform storage performance with 3D XPoint technology," *Proceedings of the IEEE*, vol. 105, no. 9, pp. 1822–1833, Sep. 2017.

[37] S. Kudekar, T. J. Richardson, and R. L. Urbanke, "Threshold saturation via spatial coupling: Why convolutional LDPC ensembles perform so well over the BEC," *IEEE Trans. Information Theory*, vol. 57, no. 2, pp. 803–834, Feb. 2011.

[38] S. Kumar, A. J. Young, N. Macris, and H. D. Pfister, "Threshold saturation for spatially coupled LDPC and LDGM codes on BMS channels," *IEEE Trans. Information Theory*, vol. 60, no. 12, pp. 7389–7415, Dec. 2014.

[39] P. M. Olmos and R. L. Urbanke, "A scaling law to predict the finite-length performance of spatially-coupled LDPC codes," *IEEE Trans. Information Theory*, vol. 61, no. 6, pp. 3164–3184, 2015.

[40] H. Esfahanizadeh, A. Hareedy, and L. Dolecek, "Finite-length construction of high performance spatially-coupled codes via optimized partitioning and lifting," *IEEE Trans. Communications*, vol. 67, no. 1, pp. 3–16, Jan. 2018.

[41] A. Hareedy, R. Wu, and L. Dolecek, "A channel-aware combinatorial approach to design high performance spatially-coupled codes," *IEEE Trans. Information Theory*, vol. 66, no. 8, pp. 4834–4852, Aug. 2020.

[42] L. Dolecek, Z. Zhang, V. Anantharam, M. J. Wainwright, and B. Nikolic, "Analysis of absorbing sets and fully absorbing sets of array-based LDPC codes," *IEEE Trans. Information Theory*, vol. 56, no. 1, pp. 181–201, Jan. 2010.

[43] S. Naseri and A. H. Banihashemi, "Spatially coupled LDPC codes with small constraint length and low error floor," *IEEE Communications Letters*, vol. 24, no. 2, pp. 254–258, Feb. 2020.

[44] S. Naseri and A. H. Banihashemi, "Construction of time invariant spatially coupled ldpc codes free of small trapping sets," *IEEE Transactions on Communications*, vol. 69, no. 6, pp. 3485–3501, Jun. 2021.

[45] M. Battaglioni, A. Tasdighi, G. Cancellieri, F. Chiaraluce, and M. Baldi, "Design and analysis of time-invariant SC-LDPC convolutional codes with small constraint length," *IEEE Trans. Communications*, vol. 66, no. 3, pp. 918–931, Mar. 2018.

[46] A. Beemer, S. Habib, C. A. Kelley, and J. Kliewer, "A generalized algebraic approach to optimizing SC-LDPC codes," in *2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2017, pp. 672–679.

[47] S. Mo, L. Chen, D. J. Costello, D. G. M. Mitchell, R. Smarandache, and J. Qiu, "Designing protograph-based quasi-cyclic spatially coupled LDPC codes with large girth," *IEEE Transactions on Communications*, vol. 68, no. 9, pp. 5326–5337, 2020.

[48] C. H. Bennett and G. Brassard, "Quantum cryptography: Public key distribution and coin tossing," in *Proceedings of IEEE International Conference on Computers, Systems, and Signal Processing*, 1984, p. 175.

[49] H.-K. Lo, M. Curty, and K. Tamaki, "Secure quantum key distribution," *Nature Photonics*, vol. 8, no. 8, p. 595, 2014.

[50] E. Diamanti, H.-K. Lo, B. Qi, and Z. Yuan, "Practical challenges in quantum key distribution," *npj Quantum Information*, vol. 2, p. 16025, 2016.

[51] Q. Zhuang, Z. Zhang, J. Dove, F. N. Wong, and J. H. Shapiro, "Floodlight quantum key distribution: A practical route to gigabit-per-second secret-key rates," *Physical Review A*, vol. 94, no. 1, p. 012322, 2016.

[52] Z. Zhang, Q. Zhuang, F. N. Wong, and J. H. Shapiro, "Floodlight quantum key distribution: Demonstrating a framework for high-rate secure communication," *Physical Review A*, vol. 95, no. 1, p. 012332, 2017.

[53] Z. Zhang, C. Chen, Q. Zhuang, F. N. Wong, and J. H. Shapiro, "Experimental quantum key distribution at 1.3 gigabit-per-second secret-key rate over a 10 db loss channel," *Quantum Science and Technology*, vol. 3, no. 2, p. 025007, 2018.

[54] X.-Q. Jiang, S. Yang, P. Huang, and G. Zeng, "High-speed reconciliation for CVQKD based on spatially coupled LDPC codes," *IEEE Photonics Journal*, vol. 10, no. 4, pp. 1–10, 2018.

[55] S. J. Johnson, V. A. Chandrasetty, and A. M. Lance, "Repeat-accumulate codes for reconciliation in continuous variable quantum key distribution," in *2016 Australian Communications Theory Workshop (AusCTW)*. IEEE, 2016, pp. 18–23.

[56] K. Zhang, X.-Q. Jiang, Y. Feng, R. Qiu, and E. Bai, "High efficiency continuous-variable quantum key distribution based on atsc 3.0 ldpc codes," *Entropy*, vol. 22, no. 10, 2020.

[57] S. Yang, A. Hareedy, S. Venkatasubramanian, R. Calderbank, and L. Dolecek, "GRADE-AO: Towards near-optimal spatially-coupled codes with high memories," in *2021 IEEE International Symposium on Information Theory (ISIT)*, Jul. 2021, pp. 587–592.

[58] S. Yang, A. Hareedy, R. Calderbank, and L. Dolecek, "Breaking the computational bottleneck: Design of near-optimal high-memory spatially-coupled codes," *arXiv preprint arXiv:2109.08978*, 2021.

[59] S. Yang, M. C. Sarihan, K.-C. Chang, C. W. Wong, and L. Dolecek, "Efficient information reconciliation for energy-time entanglement quantum key distribution," in *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, Nov. 2019, pp. 1364–1368.

[60] B. Mao, S. Wu, and H. Jiang, "Improving storage availability in cloud-of-clouds with hybrid redundant data distribution," in *2015 IEEE International Parallel and Distributed Processing Symposium*. IEEE, 2015, pp. 633–642.

[61] J. Van Lint and R. Wilson, "On the minimum distance of cyclic codes," *IEEE Transactions on Information Theory*, vol. 32, no. 1, pp. 23–40, Jan. 1986.

[62] J. Bloemer, M. Kalfane, R. Karp, M. Karpinski, M. Luby, and D. Zuckerman, "An XOR-based erasure-resilient coding scheme," 1995.

[63] J. S. Plank and L. Xu, "Optimizing Cauchy Reed-Solomon codes for fault-tolerant network storage applications," in *5th IEEE International Symposium on Network Computing and Applications (NCA'06)*, Jul. 2006, pp. 173–180.

[64] S. Wu, Y. Xu, Y. Li, and Z. Yang, "I/O-efficient scaling schemes for distributed storage systems with CRS codes," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 9, pp. 2639–2652, Sep. 2015.

[65] Z. Kong, S. A. Aly, and E. Soljanin, "Decentralized coding algorithms for distributed storage in wireless sensor networks," *IEEE Journal on Selected Areas in Communications*, vol. 28, no. 2, pp. 261–267, Feb. 2010.

[66] M. Ye and A. Barg, "Cooperative repair: Constructions of optimal MDS codes for all admissible parameters," *IEEE Transactions on Information Theory*, vol. 65, no. 3, pp. 1639–1656, Mar. 2018.

[67] "PixelExperience scales up software distribution with storj DCS." [Online]. Available: https://storj.io/documents/storj-case-study-pixel-experience.pdf

[68] S. Underwood, "Blockchain beyond bitcoin," *Communications of the ACM*, no. 11, Nov. 2016.

[69] A. R. Iyengar, P. H. Siegel, R. L. Urbanke, and J. K. Wolf, "Windowed decoding of spatially coupled codes," *IEEE Trans. Information Theory*, vol. 59, no. 4, pp. 2277–2292, Apr. 2013.

[70] D. G. M. Mitchell, M. Lentmaier, and D. J. Costello, "Spatially coupled LDPC codes constructed from protographs," *IEEE Trans. Information Theory*, vol. 61, no. 9, pp. 4866–4889, Sep. 2015.

[71] A. E. Pusane, R. Smarandache, P. O. Vontobel, and D. J. Costello, "Deriving good LDPC convolutional codes from LDPC block codes," *IEEE Trans. Information Theory*, vol. 57, no. 2, pp. 835–857, Feb. 2011.

[72] A. Hareedy, C. Lanka, and L. Dolecek, "A general non-binary LDPC code optimization framework suitable for dense flash memory and magnetic storage," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 9, pp. 2402–2415, Sep. 2016.

[73] 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," *IEEE Transactions on Information Theory*, vol. 65, no. 4, pp. 2128–2154, Apr. 2019.

[74] M. P. C. Fossorier, "Quasicyclic low-density parity-check codes from circulant permutation matrices," *IEEE Trans. Information Theory*, vol. 50, no. 8, pp. 1788–1793, Aug. 2004.

[75] L. Schmalen, V. Aref, and F. Jardel, "Non-uniformly coupled LDPC codes: Better thresholds, smaller rate-loss, and less complexity," in *2017 IEEE International Symposium on Information Theory (ISIT)*, Jun. 2017, pp. 376–380.

[76] Y. Wang, J. Yedidia, and S. Draper, "Construction of high-girth QC-LDPC codes," in *2008 5th International Symposium on Turbo Codes and Related Topics*, Sep. 2008, pp. 180–185.

[77] I. E. Bocharova, F. Hug, R. Johannesson, B. D. Kudryashov, and R. V. Satyukov, "Searching for voltage graph-based LDPC tailbiting codes with large girth," *IEEE Transactions on Information Theory*, vol. 58, no. 4, pp. 2265–2279, Apr. 2012.

[78] A. Tasdighi, A. H. Banihashemi, and M.-R. Sadeghi, "Efficient search of girth-optimal QC-LDPC codes," *IEEE Transactions on Information Theory*, vol. 62, no. 4, pp. 1552–1564, Apr. 2016.

[79] J. Wang, L. Dolecek, and R. D. Wesel, "The cycle consistency matrix approach to absorbing sets in separable circulant-based LDPC codes," *IEEE Transactions on Information Theory*, vol. 59, no. 4, pp. 2293–2314, Apr. 2013.

[80] B. Amiri, J. Kliewer, and L. Dolecek, "Analysis and enumeration of absorbing sets for non-binary graph-based codes," *IEEE transactions on communications*, vol. 62, no. 2, pp. 398–409, Feb. 2014.

[81] A. Hareedy, R. Kuditipudi, and R. Calderbank, "Minimizing the number of detrimental objects in multi-dimensional graph-based codes," *IEEE Transactions on Communications*, vol. 68, no. 9, pp. 5299–5312, Sep. 2020.

[82] 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 Transactions on Communications*, vol. 64, no. 8, pp. 3194–3207, Aug. 2016.

[83] R. Wood, M. Williams, A. Kavcic, and J. Miles, "The feasibility of magnetic recording at 10 terabits per square inch on conventional media," *IEEE Transactions on Magnetics*, vol. 45, no. 2, pp. 917–923, Feb. 2009.

[84] T. Parnell, N. Papandreou, T. Mittelholzer, and H. Pozidis, "Modelling of the threshold voltage distributions of sub-20nm NAND flash memory," in *2014 IEEE Global Communications Conference*, 2014, pp. 2351–2356.

[85] D. Declercq and M. Fossorier, "Decoding algorithms for nonbinary LDPC codes over GF(q)," *IEEE Transactions on Communications*, vol. 55, no. 4, pp. 633–643, Apr. 2007.

[86] L. Dolecek, D. Divsalar, Y. Sun, and B. Amiri, "Non-binary protograph-based ldpc codes: Enumerators, analysis, and designs," *IEEE transactions on information theory*, vol. 60, no. 7, pp. 3913–3941, Jul. 2014.

[87] A. Bazarsky, N. Presman, and S. Litsyn, "Design of non-binary quasi-cyclic LDPC codes by ACE optimization," in *2013 IEEE Information Theory Workshop (ITW)*, Aug. 2013, pp. 1–5.

[88] S. G. Srinivasa, Y. Chen, and S. Dahandeh, "A communication-theoretic framework for 2-D MR channel modeling: Performance evaluation of coding and signal processing methods," *IEEE Transactions on Magnetics*, vol. 50, no. 3, pp. 6–12, Mar. 2014.

[89] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate (corresp.)," *IEEE Transactions on Information Theory*, vol. 20, no. 2, pp. 284–287, Feb. 1974.

[90] J. Moon and J. Park, "Pattern-dependent noise prediction in signal-dependent noise," *IEEE Journal on Selected Areas in Communications*, vol. 19, no. 4, pp. 730–743, Apr. 2001.

[91] T. Zhong, H. Zhou, R. D. Horansky, C. Lee, V. B. Verma, A. E. Lita, A. Restelli, J. C. Bienfang, R. P. Mirin, T. Gerrits, S. W. Nam, F. Marsili, M. D. Shaw, Z. Zhang, L. Wang, D. Englund, G. W. Wornell, J. H. Shapiro, and F. N. C. Wong, "Photon-efficient quantum key distribution using time–energy entanglement with high-dimensional encoding," *New Journal of Physics*, vol. 17, no. 2, p. 022002, 2015.

[92] Z. Zhang, J. Mower, D. Englund, F. N. C. Wong, and J. H. Shapiro, "Unconditional Security of Time-Energy Entanglement Quantum Key Distribution Using Dual-Basis Interferometry," *Physical Review Letters*, vol. 112, no. 12, p. 120506, 2014.

[93] S. Pirandola, R. Laurenza, C. Ottaviani, and L. Banchi, "Fundamental limits of repeaterless quantum communications," *Nature Communications*, vol. 8, p. 15043, 2017.

[94] Z. Xie, T. Zhong, S. Shrestha, X. Xu, J. Liang, Y.-X. Gong, J. C. Bienfang, A. Restelli, J. H. Shapiro, F. N. Wong, and C. W. Wong, "Harnessing high-dimensional hyperentanglement through a biphoton frequency comb," *Nature Photonics*, vol. 9, no. 8, p. 536, 2015.

[95] K.-C. Chang, X. Cheng, M. C. Sarihan, A. K. Vinod, Y. S. Lee, T. Zhong, Y.-X. Gong, Z. Xie, J. H. Shapiro, F. N. Wong *et al.*, "648 hilbert-space dimensionality in a biphoton

frequency comb: entanglement of formation and schmidt mode decomposition," *npj Quantum Information*, vol. 7, no. 1, pp. 1–11, 2021.

[96] I. Ali-Khan, C. J. Broadbent, and J. C. Howell, "Large-alphabet quantum key distribution using energy-time entangled bipartite states," *Phys. Rev. Lett.*, vol. 98, p. 060503, Feb 2007. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.98.060503

[97] N. J. Cerf, M. Bourennane, A. Karlsson, and N. Gisin, "Security of quantum key distribution using d-level systems," *Physical Review Letters*, vol. 88, no. 12, p. 127902, 2002.

[98] L. Sheridan and V. Scarani, "Security proof for quantum key distribution using qudit systems," *Physical Review A*, vol. 82, no. 3, p. 030301, 2010.

[99] C. Lee, J. Mower, Z. Zhang, J. H. Shapiro, and D. Englund, "Finite-key analysis of high-dimensional time–energy entanglement-based quantum key distribution," *Quantum Information Processing*, vol. 14, no. 3, pp. 1005–1015, 2015.

[100] E. Karimi, E. Soljanin, and P. Whiting, "Increasing the raw key rate in energy-time entanglement based quantum key distribution," in *2020 54th Asilomar Conference on Signals, Systems, and Computers*, 2020, pp. 433–438.

[101] C. Lee, D. Bunandar, Z. Zhang, G. R. Steinbrecher, P. B. Dixon, F. N. Wong, J. H. Shapiro, S. A. Hamilton, and D. Englund, "Large-alphabet encoding for higher-rate quantum key distribution," *Optics express*, vol. 27, no. 13, pp. 17 539–17 549, 2019.

[102] M. C. Sarihan, K.-C. Chang, X. Cheng, Y. S. Lee, C. Chen, T. Zhong, H. Zhou, Z. Zhang, F. N. Wong, J. H. Shapiro *et al.*, "Frequency-multiplexed rate-adaptive quantum key distribution with high-dimensional encoding," in *CLEO: QELS_Fundamental Science*. Optical Society of America, 2020, pp. FF3C–3.

[103] M. C. Sarihan, K.-C. Chang, X. Cheng, H. Tsuda, and C. W. Wong, "Proof-of-principle frequency-bin quantum key distribution with biphoton frequency combs," in *CLEO: Applications and Technology*. Optical Society of America, 2021, pp. ATu1S–6.

[104] T. Richardson, R. Urbanke *et al.*, "Multi-edge type ldpc codes," in *Workshop honoring Prof. Bob McEliece on his 60th birthday, California Institute of Technology, Pasadena, California*, 2002, pp. 24–25.

[105] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Information Theory*, vol. 47, no. 2, pp. 619–637, 2001.

[106] H. Jin, A. Khandekar, R. McEliece *et al.*, "Irregular repeat-accumulate codes," in *Proc. 2nd Int. Symp. Turbo codes and related topics*. Citeseer, 2000, pp. 1–8.

[107] A. Roumy, S. Guemghar, G. Caire, and S. Verdú, "Design methods for irregular repeat-accumulate codes," *IEEE Transactions on Information Theory*, vol. 50, no. 8, pp. 1711–1727, 2004.

[108] G. Yue, X. Wang, and M. Madihian, "Design of rate-compatible irregular repeat accumulate codes," *IEEE Transactions on Communications*, vol. 55, no. 6, pp. 1153–1163, 2007.

[109] T. Zhong, F. N. Wong, T. D. Roberts, and P. Battle, "High performance photon-pair source based on a fiber-coupled periodically poled ktiopo 4 waveguide," *Optics express*, vol. 17, no. 14, pp. 12 019–12 030, 2009.

[110] T. Zhong, X. Hu, F. N. Wong, K. K. Berggren, T. D. Roberts, and P. Battle, "High-quality fiber-optic polarization entanglement distribution at 1.3 μm telecom wavelength," *Optics Letters*, vol. 35, no. 9, pp. 1392–1394, 2010.