Distributed Multi-User Secret Sharing and Results on Unique and List Decoding of Codes

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science in Electronics and Communication Engineering by Research

by

Rasagna Chigullapally
20171003
rasagna.c@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
November 2023

Copyright © Rasagna Chigullapally, 2023
All Rights Reserved

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "Distributed Multi-User Secret Sharing and Results on Unique and List Decoding of Codes" by Rasagna Chigullapally, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Advisor: Dr. Lalitha Vadlamani

To my Family and Friends

Acknowledgments

I would like to express my heartfelt gratitude to my advisor, Dr. Lalitha Vadlamani, for her invaluable guidance and unwavering support throughout my research journey. Despite the additional time we required for our research, she remained patient, encouraging, and understanding, providing a nurturing environment for learning and growth. I am really grateful to my co-advisor Dr. Prasad Krishnan for his teaching in the beginning years and guidance in the later years of MS. I would also like to thank Dr. Nikhil Karamchandani (IIT Bombay) for his guidance and support, and also Myna Vajha for her teaching and guidance in the starting stages of my research. Next, I would like to thank my supportive, patient, and brilliant research partner Harshithanjani Athi.

I would like to thank the Qualcomm Innovation Fellowship 2022 and the mentors Ashutosh Gore and Ahmed Zaki for their input and encouragement throughout the fellowship.

I am deeply appreciative of my parents and brother for their unwavering support throughout my college journey. Their belief in me and trust in my decisions have been invaluable. I want to express my heartfelt gratitude for always standing by my side and reassuring me that I can rely on them whenever I need assistance. Your love and encouragement have played a significant role in shaping who I am today, and I am forever thankful for your presence in my life.

To Lingam, and my little brother Annam, I would like to thank you guys for being my constant companions throughout my college years and Teru and Siri for all our random late-night talks, walks, and your support and help. My college life is filled with memories with you guys, and I can't imagine it without you guys. I would like to thank my silly roommate Shivani. I would like to thank my friends Harshitha, Meghana, Varsha, Amul, Yippie, Rupa, Anudeep, Gayathri, Thejasvi, Chintoo, Bheemala, Aravind, Jayadev, PV, Sushanth, Bijjam. I would like to thank all my dual degree friends for all the support and fun in the 5th year. Lastly, I would like to thank all the people I have interacted with throughout my college life for making it a pleasant memory.

Abstract

In this thesis, we consider three problems relating to coding theory. The first problem is the application of codes to cryptography, while the other two problems tackle the ongoing developments of coding theory.

We consider the secret sharing problem in the context of distributed storage. In this setup, there is a dealer, n storage nodes, m secrets, and $\binom{m}{t}$ users. The dealer wants to share with each user a t-subset of secrets via storage nodes while ensuring *weak secrecy* condition. The user downloads shares from the storage nodes based on the designed storage structure and reconstructs its secrets. We extend upon a recent framework [1] (considering $t = 1$) to the case where each user requests a subset of secrets. We identify a necessary condition on the storage structure to ensure *weak secrecy*. A distributed secret sharing protocol (DSSP) encodes secrets into shares and distributes them onto the storage node. The number of shares created per secret is defined as the storage overhead of the DSSP, which is at least one to ensure the correctness condition. The goal is to design a DSSP with optimal storage overhead. We relate the storage structure of a DSSP to t-disjunct matrices used in group testing. Using a storage structure obtained from disjunct matrices, we design a weakly secure DSSP that achieves optimal storage overhead. Using several constructions for t-disjunct matrices, we compare their performance in terms of the number of storage nodes and communication complexity (total number of shares needed to be downloaded from storage nodes). The rate of a secret is the size of the secret normalized by the size of a share, and the capacity region of the problem is the set of all achievable rate tuples. For a specified storage structure, we characterize the capacity region of the problem.

Next, we consider the unique decoding algorithms for Reed-Muller (RM) codes. RM codes were shown to be capacity-achieving on the binary memoryless symmetric channels using bitwise maximum-a-posteriori (bit MAP) decoding [2]. We propose an RXA-Syndrome decoding algorithm for $RM(m, m-4)$ code that exploits the large symmetry in the automorphism group of RM codes and the optimality of the maximum likelihood (ML) decoder.

Codes that meet the Singleton bound with equality are called Maximum Distance Separable (MDS) codes. In a sense, MDS codes achieve the optimal trade-off between the size of the code and its minimum distance. However, unique decoding of MDS codes can correct errors only up to a certain limit which is related to the minimum distance of the code. In the third problem, we consider the construction of codes that can correct errors beyond this limit. To this end, list decoding was introduced recently, which returns a list of codewords containing the correct codeword. We derive necessary and sufficient

conditions for a code to be (t, L)-list decodable. Recently, list decodable codes that meet generalized Singleton bound were considered as the higher-order generalization of MDS codes. It is important to understand the structural properties of these codes to develop explicit constructions and efficient decoding algorithms. Motivated by these developments, we present new results on the structure of higher order MDS codes. We identify conditions under which (n_1, k_1)-MDS(ℓ_1) codes can be obtained from (n_2, k_2)-MDS(ℓ_2) codes. We also simplify the conditions under which an (n, k)-MDS$(k-1)$ code is also an MDS(k) code.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Coding theory focuses on the design and analysis of error-detecting and error-correcting codes. The main goal is to develop efficient and reliable techniques for transmitting data while ensuring low redundancy and ease of error detection or correction. It is an indispensable tool in Data Compression, DNA Storage Systems, Cryptography, Group Testing, Distributed Storage, Private Information Retrieval, etc. It has recently been used in Quantum Information, Federated Learning, and Machine Learning.

Modern cryptography deals with the construction and analysis of the protocols for securing digital information, which includes secure communication, authentication, digital signatures, secret sharing, and distributed elections, to name a few. The primary goal is to enable secure transmission of information between trusted parties over an untrusted channel.

A secret sharing scheme is an essential tool in cryptography that enables secure communication among untrusted agents, enhances fault tolerance against partial loss of information, and facilitates key management in key-based encryption systems. A secret sharing scheme defines an access structure determining which subset of users can reconstruct the secret. Section 1.1 provides a literature review and our contributions to this problem.

The problem of decoding the received message sent over a noisy communication channel is of particular interest in coding theory. The unique decoding is one such decoding technique that picks the unique codeword that is closest to the received message. We investigate the problem of designing efficient unique decoding algorithms for a special class of codes called Reed-Muller (RM) codes. The maximum likelihood decoding algorithm turns out to be impractical for general parameters of RM code. However, RM codes have large automorphism groups that induce symmetry. Recent works have shown that carefully exploiting this symmetry is the key to designing efficient algorithms. Section 1.2 presents a literature review and our contributions in this regard.

The unique decoding can be shown to correct any error patterns given that the number of errors is at most $\lfloor \frac{d-1}{2} \rfloor$, where d is the minimum distance of the code. To correct errors beyond this limit, list decoding was introduced, which returns a list of possible candidate codewords containing the correct codeword. This opened a plethora of questions related to list decodable codes and their structural properties. A literature review and our contribution to this account are provided in Section 1.3.

1.1 Distributed Multi-User Secret Sharing with Multiple Secrets per User

A secret sharing scheme is a cryptographic technique where a dealer distributes a secret among multiple users while maintaining two key properties: secret recovery, which allows authorized subsets of parties to reconstruct the secret from their shares, and collusion resistance, which ensures that unauthorized subsets of parties cannot learn anything about the secret. These properties are crucial in maintaining the confidentiality and integrity of the secret.

The concept of secret sharing was first introduced by Shamir [3] and Blakley [4] in their independent works. In [3], Shamir proposes a secret sharing scheme based on polynomial interpolation, while in [4], Blakley introduces a secret sharing scheme based on the intersection of subspaces. Secret sharing has been extensively studied and applied in various areas of cryptography and distributed computing, such as secure multiparty computation [5], Byzantine agreement [6], secure cloud computing [7], threshold cryptography [8], attribute-based encryption [9], secure voting systems [10], to name a few. Moreover, recent advances in secret sharing have enabled its use in emerging technologies such as blockchain [11] and secure multi-party machine learning [12].

The increasing prevalence of distributed storage systems in recent years has sparked the need to extend secret sharing schemes to accommodate such setups. In the distributed storage setup, the dealer has control over a set of storage nodes that are used to save the shares of the secret. This extension is first considered in [1], where it is supposed that each user requests a distinct secret and is therefore given access to a certain subset of storage nodes. It is referred to as distributed multi-user secret sharing (DMUSS) problem. There are two privacy conditions that are considered: *weak secrecy* and *perfect secrecy*, which quantify the amount of information that can be known about other users' secrets given the shares of a specific user. The storage overhead and communication complexity are identified to be the key parameters of a distributed secret sharing protocol (DSSP). Several DSSPs are constructed in [1] that achieve the optimal values for these parameters. In [13], the notion of the rate of a secret is introduced. For a DMUSS, the set of achievable rate tuples is defined to be its capacity region. The capacity region is characterized by a convex hull of rate tuples satisfying certain inequalities.

We extend the DMUSS problem to allow users to request a subset of secrets instead of a single secret. The contributions are as follows:

- We derive a necessary condition on the storage structure of the distributed secret sharing protocol to ensure weak secrecy and establish a relation between the storage structure and the t-disjunct matrices.

- Using the storage structure obtained from the t-disjunct matrix, we propose a secret sharing protocol that achieves optimal storage overhead.

- Using several constructions for t-disjunct matrices, we compare the system parameters and properties. We also show that t-disjunct matrices obtained using the Steiner system are better than those obtained from other known constructions in terms of accommodating more secrets.

- We characterize the capacity region of the distributed multi-user secret sharing system when the storage structure is specified.

1.2 Decoding Reed-Muller Codes

Recently, Reed–Muller codes were shown to achieve capacity on the binary memoryless symmetric channels using bit MAP decoding in certain parameter regimes [2]. This re-ignited the interest in developing efficient decoding algorithms.

Reed–Muller (RM) code with parameters m, r, $RM(m, r)$, is the set of 2^m-length evaluation vectors of all binary m-variate polynomials in $\mathbb{F}_2$ with degree at most r. Reed in [14] was the first to propose a decoding algorithm based on majority voting logic which can correct any error pattern up to $\frac{d}{2}$ errors for a general m, r in $O(n \log^r n), n = 2^m$. For the first-order RM codes, fast Hadamard transform implementation of maximum likelihood (ML) decoder results in $O(n \log n)$ complexity [15].

The optimal ML decoder is impractical for general parameter regimes due to its exponential complexity, which led to the design of efficient decoding algorithms which approach ML performance. In [16], a Soft-Decision decoding algorithm is proposed, which builds on the recursive definition of RM codes. Any codeword $c \in RM(m, r)$ can be written as $(u, u + v)$ where $u \in RM(m - 1, r)$ and $v \in RM(m - 1, r - 1)$. Using this fact, the problem is broken into smaller sub-problems until a base case of $r = 1$ or $m = r$ is attained, which can be efficiently solved using ML decoding. These solutions are combined using the recursive definition to reconstruct the solution at the next level and so on. The Soft-Decision decoding algorithm has $O(n \log n)$ complexity while approaching ML performance.

Exploiting the vast symmetry present in the automorphism group of RM code has led to several efficient decoding algorithms. In [17], Recursive Projection-Aggregation (RPA) decoding is proposed, which uses all possible permutations from the automorphism group to generate new codewords of the received vector, which is then recursively shortened by one order using projection. To construct the solution of the next level, an aggregation of 2^m codewords is performed to ensure correctness. RPA has $O(n^r \log n)$ complexity, which can be further reduced to $O(n^2)$ with $O(n^r)$ parallel processors. Recursive Puncturing-Aggregation (RXA) decoding is proposed in [18] where recursion is performed on both halves of the codeword. Both these algorithms work for binary symmetric channels and binary-input additive white Gaussian noise channels. In [19], hard-decision ML syndrome decoding of $RM(m, m-3)$ code is considered, which works for a binary symmetric channel. The permutations are used to transform the syndrome to lie in a reduced syndrome table, thus ensuring an overall decoding complexity of $O(m^3)$. We propose an RXA-Syndrome decoding algorithm for $RM(m, m - 4)$ code. We also present simulation results comparing the decoding performance of the proposed algorithm against the RPA algorithm, which is known to perform close to the ML decoder in some parameter regimes.

1.3 Structural Properties of Higher-Order MDS codes

Singleton bound characterizes the trade-off between the size of the code and its minimum distance, which is closely related to the number of errors that can be corrected for a code. Codes that meet Singleton bound with equality are called maximum distance separable (MDS) codes. Thus, MDS codes achieve this optimal trade-off and consequently correct more errors compared to any other code for a fixed message and block lengths.

List decoding can correct more errors than $\lfloor \frac{d-1}{2} \rfloor$ owing to the relaxation of returning a list of candidate solutions rather than a unique codeword. Generalized Singleton bound is introduced in [20] for list decodable codes, which gives an upper bound on the size of the code. The natural question is to construct codes that meet the bound with equality. A stronger notion of list decodability is considered in [21] as a generalization to MDS codes, i.e., average-radius list decodable codes that meet the generalized Singleton bound with equality, are defined to be list decodable maximum distance separable (LD-MDS) codes.

Using the notion of generic collections, [22] introduced another combinatorial extension of MDS codes as higher-order MDS (MDS(ℓ)) codes. A duality relation connects LD-MDS to MDS(ℓ) codes, as shown in [22]. The Reed-Solomon (RS) codes are shown to achieve the generalized Singleton bound for a large enough field size.

This leaves many open questions relating to the structure of higher order MDS codes and bounds on field sizes. Our contributions are as follows:

- We present necessary and sufficient conditions for list decodability.
- We show that an (n, k)-MDS(k) code is also (n, k)-MDS(ℓ) for all $\ell > k$.
- We deduce simpler necessary conditions for an (n, k)-MDS($k-1$) code to also qualify as an MDS(k) code.

1.4 Organization

The thesis is organized as follows. In Chapter 2, we provide the background material for the content that follows. Chapter 3 details our results on the distributed multi-user secret sharing problem. Chapter 4 discusses the decoding algorithms for RM codes. In Chapter 5, we discuss the structural results on higher-order MDS codes, and Chapter 6 ends the thesis with some concluding remarks.

Chapter 2

Background

In this chapter, we introduce the necessary background material for the rest of the thesis. In Section 2.1, we discuss the secret sharing problem and present Shamir's secret sharing scheme. In Section 2.2, we generalize the secret sharing problem to distributed storage setup and mention the key differences in this setting. In Section 2.3, we present some terms and results in coding theory. Lastly, in Section 2.4, we introduce disjunct matrices and present several well-known disjunct matrix constructions.

Notation

For $n \in \mathbb{N}$, define $[n]$ as the set $\{1, 2, \ldots, n\}$, $2^{[n]}$ denotes the power set of $[n]$ and for $n_1, n_2 \in \mathbb{N} \cup \{0\}$, $n_1 \leq n_2$, define $[n_1 : n_2]$ as the set of $\{n_1, n_1 + 1, \ldots, n_2\}$. For a set $\mathcal{I} = \{i_1, \ldots, i_n\}$ and $C = \{C_1, \ldots, C_m\}$, $C_{\mathcal{I}}$ represents the set $\{C_{i_1}, \ldots, C_{i_n}\}$. For an $m \times n$ matrix A, the rank of the matrix A is the number of linearly independent columns, and A is said to be full rank if $\mathsf{rk}(A) = \min\{m, n\}$.

2.1 Secret Sharing

We consider the problem where a dealer wants to securely share a secret s with n participating users. To this end, the dealer encodes the secret into shares and distributes them to the users such that only authorized users can reconstruct the secret, and unauthorized users get no information about the secret from their shares. We formalize this notion as the secret sharing scheme, given below.

Definition 2.1. *A secret sharing scheme is a bundle of ($\mathcal{A}$, $\mathcal{E}$, $\mathcal{D}$) defined below.*

- *The collection $\mathcal{A} \subseteq 2^{[n]}$ is called the access structure such that if $B \in \mathcal{A}$ and $B \subseteq C \subseteq [n]$, then $C \in \mathcal{A}$. We call the sets in $\mathcal{A}$ as authorized and others as unauthorized.*
- *The function $\mathcal{E} : \mathbb{F}_q \to \mathbb{F}_q^n$ is called an encoder that takes a secret s and returns n shares to be distributed to the users which we represent as $\mathbf{y} = (y_1, \ldots, y_n)^T = \mathcal{E}(s)$. For each $B \subseteq [n]$, let $\mathbf{y}_B$ denote the vector of shares distributed to the set of users given by B.*

- *The function $\mathcal{D} : \mathbb{F}_q^{\leq n} \to \mathbb{F}_q$ is called a decoder, such that for all $B \in \mathcal{A}$, $\mathcal{D}(\mathbf{y}_B) = s$, i.e., each authorized set of users can successfully reconstruct the secret. This is referred to as correctness condition.*

- *Every unauthorized set of users, when combining their shares gain nothing about the secret (in the information-theoretic sense), i.e., for all $C \notin \mathcal{A}$, $H(s|\mathbf{y}_C) = H(s)$, where H is the entropy. This is referred to as perfect secrecy condition.*

Remark 2.2. *For the sake of brevity, in perfect secrecy condition, we use entropy for a particular secret s to denote the entropy of the random variable itself. We adopt this notation for the content that follows.*

Definition 2.3. *A (t, n)-thresholding scheme is a secret sharing scheme in which the access structure is defined as follows:*

$$\mathcal{A} = \{B \subseteq [n] \ : \ |B| \geq t\}. \tag{2.1}$$

Next, we present Shamir's secret sharing scheme in some detail for the sake of completeness.

2.1.1 Shamir's Scheme

Suppose $s \in \mathbb{F}_q$ is the secret to be shared.

Assumption 2.4. *The secret s is uniformly distributed in the domain of secrets.*

Algorithm 1 The (t, n) Shamir's Scheme

- **(Initialization)** Pick n distinct non-zero elements $x_1, \ldots, x_n \in \mathbb{F}_q$ which are made public.

- **(Share Distribution)** Construct the polynomial $p(x) = s + \sum_{k=1}^{t-1} a_k x^k$ by picking $t-1$ random elements $a_1, \ldots, a_{t-1} \in \mathbb{F}_q$. The shares are computed by polynomial evaluation: $y_i = p(x_i)$ for $1 \leq i \leq n$. Each share y_i is given to the user i.

Remark 2.5. *We refer to the encoder $\mathcal{E}$ that takes a secret s as the input and outputs $(y_1, \ldots, y_n)$ as described above as a (t, n)-Shamir's encoder.*

Proposition 2.6. *The (t, n) Shamir's scheme described above is a (t, n) thresholding scheme.*

Proof. We need to prove that the correctness and perfect secrecy conditions hold when the access structure $\mathcal{A}$ is a collection of all sets with cardinality at least t.

(Correctness) Suppose $B \in \mathcal{A}$ and $|B| = \ell \geq t$. WLOG assume that $B = [t]$. Then for each $1 \leq i \leq t$, we know $y_i = p(x_i)$ where $p(x) = \sum_{k=0}^{t-1} a_k x^k$ and a_k's unknown. The corresponding linear system of equations can be compactly written as:

$$\begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{t-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{t-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_\ell & x_\ell^2 & \cdots & x_\ell^{t-1} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{t-1} \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_\ell \end{bmatrix}. \tag{2.2}$$

The left-hand side matrix is the Vandermonde matrix. It has full rank if and only if x_i's are all distinct, which is true by the initialization step. Consequently, there exists a unique solution $(a_0, \ldots, a_{t-1})^T$ for the linear system (2.2) and therefore a unique polynomial p. The secret is then obtained by computing $p(0)$. Thus, any authorized set of user B can successfully reconstruct the secret.

(Perfect secrecy) Consider a set $U = \{i_1, \ldots, i_{t-1}\}$ and let $a \in \mathbb{F}_q$. Since $|U| = t - 1$, $U \notin \mathcal{A}$. From the approach used above, notice that there exists a unique polynomial p_a such that $p_a(0) = a$ and $p_a(x_{i_k}) = y_{i_k}$ for $1 \leq k \leq t-1$. The probability that the scheme chooses this choice of non-constant coefficients is given by

$$\Pr[\mathbf{y}_U | \text{secret} = a] = \frac{1}{q^{t-1}}. \tag{2.3}$$

Note that it is independent input a, the input of the encoder. Then we have

$$\begin{aligned} \Pr[\mathbf{y}_U | \text{secret} = a] &= \Pr[\mathbf{y}_U | \text{secret} = a] \sum_{b \in \mathbb{F}_q} \Pr[\text{secret} = b] \\ &= \sum_{b \in \mathbb{F}_q} \Pr[\text{secret} = b] \Pr[\mathbf{y}_U | \text{secret} = a] \\ &= \sum_{b \in \mathbb{F}_q} \Pr[\text{secret} = b] \Pr[\mathbf{y}_U | \text{secret} = b] \qquad \text{(by (2.3))} \\ &= \Pr[\mathbf{y}_U]. \end{aligned}$$

This shows that $\mathbf{y}_U$ is independent of secret or, in other words, $H(s|\mathbf{y}_U) = H(s)$. □

Remark 2.7. *The mapping that reconstructs the secret s given an authorized set $B \in \mathcal{A}$ as described in the correctness proof is referred to as (t, n)-Shamir's decoder.*

2.2 Distributed Multi-User Secret Sharing

Consider the distributed storage scenario where the shares generated by the dealer are not readily available to the users, and rather the dealer has control over a set of storage nodes. The users have access to a certain subset of storage nodes. Additionally, consider the multi-user setup where each user requests a different secret. This setup is referred to in [1] as the distributed multi-user secret sharing (DMUSS).

In this scenario, each user has a designated secret message and is given access to a certain subset of storage nodes, where the user can download the stored data. The goal of the dealer is to convey the designated secret to each user via storage nodes. The correctness condition translates to: each user can successfully recover the designated secret by accessing their shares in the storage nodes. Under the multi-user context, two secrecy conditions are considered depending on the amount of information that can be known about other secrets given the shares of a specific user. The *weak secrecy* condition requires that each user does not get any information about the individual secrets of other users, while the *perfect secrecy* condition requires that a user does not get any information about the collection of other users' secrets.

The notion of access structure is different in this setup. Whereas each set in the access structure is represented as a subset of authorized users to reconstruct the secret, here, it represents a subset of storage nodes that can be used to reconstruct a secret. Thus, each secret can be reconstructed by only one set in the access structure.

The storage structure of a DMUSS is the set of subsets of storage nodes, where each subset represents storage nodes in which shares of a secret are stored. We can observe that in this scenario, the access structure and storage structure are equivalent.

In the next chapter, we go into further details of DMUSS.

2.3 Some Terms and Results in Coding Theory

In this section, we introduce some terms and results in coding theory that will be used later on.

Proposition 2.8 (Singleton Bound)**.** *A $[n, k, d]_q$ linear code satisfies $k \leq n - d + 1$.*

Remark 2.9. *The Singleton bound can be rephrased in terms of the fraction of errors $\rho = \frac{1}{n}\lfloor\frac{d-1}{2}\rfloor$ as:*

$$k \leq n - d + 1 \iff \rho \leq \tfrac{1-R}{2} \iff |C| \leq q^{n(1-2\rho)}.$$

Definition 2.10 (MDS Code)**.** *An $[n, k, d]_q$ linear code that meets Singleton bound with equality, i.e., $d = n - k + 1$, is referred to as Maximum Distance Separable (MDS) code.*

Definition 2.11 (RS Code)**.** *An $[n, k]_q$ Reed-Solomon code (RS code) with $\gamma_1, \ldots, \gamma_n \in \mathbb{F}_q$ as the evaluation points, maps a message $(m_0, \ldots, m_{k-1})$ to the polynomial $f(x) = \sum_{i=0}^{k-1} m_i x^i$ evaluated at*

$\gamma_1, \ldots, \gamma_n$, *i.e.,*

$$RS : (m_0, \ldots, m_{k-1}) \mapsto (f(\gamma_1), \ldots, f(\gamma_n)).$$

Proposition 2.12. *The $[n, k]_q$ RS code is an MDS code.*

Definition 2.13 (Code Concatenation)**.** *Let C_{out} be an $[N, K, D]_{q^k}$ code and C_{in} is an $[n, k, d]_q$ code. Then given a message $m \in [q^k]^K$, the concatenated code $C_{out} \circ C_{in} : [q]^{kK} \to [q]^{nN}$ is defined as*

$$C_{out} \circ C_{in}(m) = (C_{in}(c_1), \ldots, C_{in}(c_N)), \text{ where } C_{out}(m) = (c_1, \ldots, c_N).$$

Proposition 2.14. *If C_{out} is an $[N, K, D]_{q^k}$ code and C_{in} is an $[n, k, d]_q$ code, then their concatenated code $C_{out} \circ C_{in}$ is an $[nN, kK, dD]_q$ code. The rate and minimum distance of the concatenated code are the product of the rate and minimum distance of both codes, respectively.*

Definition 2.15 (Automorphism Group)**.** *The automorphism group of a code C is defined as the set of bijective mapping (or permutation) π that map C onto itself, i.e.,*

$$\mathrm{Aut}(C) = \{\pi : C \to C \ : \ \pi \text{ is bijective}, \pi(c) \in C, \forall c \in C\}.$$

2.4 Disjunct Matrices

In Chapter 3, we will use disjunct matrices to obtain storage structures. So in this section, we describe disjunct matrices in the group testing setup, where they play a fundamental role.

2.4.1 Group Testing

Problem. *Consider a population of size m that contains at most t (unknown) defective items. Suppose the population is represented as $x \in \{0, 1\}^m$, where 1 denotes a defective element. A test can be run on a subset of the population whose result is 'positive' if there is at least one defective element in the subset and 'negative' if there is none. The group testing problem is to find all the defective elements in x using the smallest number of tests.*

Definition 2.16 (Test)**.** *A test is a subset $S \subseteq [m]$ whose result $A(S)$ is the logical-OR of all elements of x in S, i.e.,*

$$A(S) = \bigvee_{i \in S} x_i.$$

Suppose a testing method performs a sequence of n tests. It can be compactly represented as an $n \times m$ testing matrix $M = [m_{ij}]$ where

$$m_{ij} = \begin{cases} 1 & \text{if index } j \text{ is in test } i \\ 0 & \text{otherwise} \end{cases}$$

Let us represent the result of these n tests as $y = M \odot x$, where in the matrix-vector $\odot$ multiplication, the scalar addition is replaced by logical-OR and the scalar multiplication by logical-AND. Since x is a binary vector, one can rewrite the multiplication as

$$y = M \odot x = \bigcup_{i \in \text{supp}(x)} m_i,$$

where m_i is the i-th column of M, $\text{supp}(x) = \{i : x_i \neq 0\}$, and the union of columns is performed as element-wise logical-OR for each element in the column. Note that $y_i = 1$ if there exists a defective element in test i and $y_i = 0$ otherwise. In this representation, the trivial method of testing each element in a separate test would be represented by $m \times m$ identify matrix.

Definition 2.17 ($n(t, m)$)**.** *Given an input $x \in \{0,1\}^m$ with $\text{wt}(x) \leq t$, the number of tests required to successfully reconstruct x is defined as $n(t, m)$.*

Consider the parity check matrix H_k for $[2^k - 1, 2^k - k - 1, 3]_2$, where the i-th column is the binary representation of i (1-based indexing) and $2^{k-1} < m \leq 2^k - 1$. If $\text{wt}(x) \leq 1$, then $H_k \odot x$ uniquely determines x. This gives the following bound on $n(1, m)$.

Proposition 2.18. $n(1, m) = \Theta(\log m)$.

Definition 2.19 (t-separable)**.** *A $n \times m$ binary matrix M is t-separable if for every $S_1 \neq S_2 \subseteq [m]$ such that $|S_1| \leq t$ and $|S_2| \leq t$, we have*

$$\bigcup_{j \in S_1} m_j \neq \bigcup_{i \in S_2} m_i.$$

The following result shows that the test matrices M for which $M \odot x$ uniquely determining x are exactly the t-separable matrices.

Proposition 2.20. *M is t-separable if and only if for every $x_1 \neq x_2 \in \{0,1\}^n$ such that $\text{wt}(x_1) \leq t$ and $\text{wt}(x_2) \leq t$ we have*

$$M \odot x_1 \neq M \odot x_2.$$

Even though t-separable matrices capture the notion of uniquely determining x from $M \odot x$, the time complexity required for this operation is $m^{\Theta(t)}$, which scales exponentially with t.

Thus, we are led to consider a special class of t-separable matrices that have efficient decoding algorithms.

2.4.2 Disjunct Matrices

Definition 2.21 (t-disjunct)**.** *A $n \times m$ binary matrix M is t-disjunct if the union of supports of any t columns of the matrix does not contain the support of any other column, i.e., for every $S \subseteq [m]$ with $|S| \leq t$ and every $j \notin S$, we have*

$$m_j \not\subseteq \bigcup_{i \in S} m_i.$$

Proposition 2.22. *Let M be an $n \times m$ t-disjunct matrix. Then*

(a) M is also t-separable, and

(b) there exists a $\mathcal{O}(nm)$ time decoding algorithm for M.

Proposition 2.23. *Let $1 \leq t \leq m$ be an integer and M be a $n \times m$ matrix such that*

(i) every column in M has at least w_{min} ones, i.e.,

$$\forall\, j \in [m], \quad \mathrm{wt}(m_j) \geq w_{min}, \text{ and}$$

(ii) every distinct pair of columns in M has at most a_{max} ones in common, i.e.,

$$\forall\, i \neq j \in [m], \quad \mathrm{wt}(m_i \cap m_j) \leq a_{max},$$

for some integers $a_{max} \leq w_{min} \leq n$. Then M is $\left\lfloor \frac{w_{min}-1}{a_{max}} \right\rfloor$-disjunct.

Using the sufficient condition given above, we define some of the well-known coding-theoretic constructions for t-disjunct matrices.

2.4.3 Kautz-Singleton Construction [23]

A $[q, k, q-k+1]_q$ Reed-Solomon code is picked as the outer code $\mathcal{C}_{out}$ while the inner code $\mathcal{C}_{in} : \mathbb{F}_q \to \{0,1\}^q$ is defined as follows. For any $i \in \mathbb{F}_q$, $\mathcal{C}_{in}(i) = e_i$, where e_i is nothing but a one-hot vector. The concatenated code $\mathcal{C}^* = \mathcal{C}_{out} \circ \mathcal{C}_{in}$ is a $n \times m$ t-disjunct matrix, where $n = q^2, m = q^k$, and $t = \lfloor \frac{q-1}{k-1} \rfloor$. Here, each column in $\mathcal{C}^*$ has q ones.

Example 2.24. *Let $k = 2$, $q = 3$. Then the outer code, which is $[3, 2]_3$-RS code, and the inner code can be represented as the following:*

$$M_{C_{out}} = \begin{pmatrix} 0 & 1 & 2 & 0 & 1 & 2 & 0 & 1 & 2 \\ 0 & 1 & 2 & 1 & 2 & 0 & 2 & 0 & 1 \\ 0 & 1 & 2 & 2 & 0 & 1 & 1 & 2 & 0 \end{pmatrix}, \qquad M_{C_{in}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

where we arranged the codewords as the columns of the matrix. The concatenated code $C^* = C_{out} \circ C_{in}$ *is given below, where the rows in* M_{C^*} *that correspond to the same row in* $M_{C_{out}}$ *have the same color.*

$$M_{C^*} = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

2.4.4 Porat-Rothschild Construction [24]

A linear code that meets the Gilbert-Varshamov bound is picked as the outer code. Here, $\mathcal{C}_{out}$ is $[\mathscr{z}, k, \delta\mathscr{z}]_q$ linear code, where $\mathscr{z} \leq \frac{k}{1-H_q(\delta)}$ and $t+1 = \lceil \frac{1}{1-\delta} \rceil$. The inner code $\mathcal{C}_{in} : \mathbb{F}_q \to \{0,1\}^q$ is defined as follows. For any $i \in \mathbb{F}_q$, $\mathcal{C}_{in}(i) = e_i$, where e_i is nothing but a one-hot vector. The concatenated code $\mathcal{C}^* = \mathcal{C}_{out} \circ \mathcal{C}_{in}$ is a $n \times m$ t-disjunct matrix, where $n = \mathscr{z}q, m = q^k$ and $t = \lceil \frac{1}{1-\delta} \rceil - 1$. Here, each column in $\mathcal{C}^*$ has $\mathscr{z}$ ones.

2.4.5 Sparse Disjunct Matrices [25]

In a $n \times m$ Sparse disjunct matrix, the number of ones in each column of a t-disjunct matrix is restricted to $\ell t + 1$, where $\ell \geq 1$. Such a matrix can be constructed by replacing the outer code in the Kautz-Singleton construction with $[\ell t + 1, k = \ell + 1]$-RS code over a field of size $q = \sqrt[\ell+1]{m}$. The concatenated code $\mathcal{C}^*$ is a $(\ell t + 1)q \times m$ t-disjunct matrix.

2.4.6 Steiner system [26]

Definition 2.25. *Let* X *be an* n*-element set. A Steiner system* $\mathfrak{S}(n, b, p)$ *is defined as* $\mathfrak{S} \subset \binom{X}{b}$ *such that for every* $A \in \binom{X}{p}$ *there is exactly one* $B \in \mathfrak{S}$ *with* $A \subset B$*, where* $\binom{X}{i}$ *here denotes the collection of all the* i*-sized subsets of* X*. The largest set* $\mathfrak{S}$ *which satisfies this property is called the maximum Steiner system.*

Example 2.26. *The Steiner system* $\mathfrak{S}(n, 3, 2)$ *is called Steiner triple system, which is shown to exist for* $n \geq 7$ *and* $n \equiv 1$ *or* $3 \pmod 6$ *[27]. Suppose* $n = 9$ *and* $X = [9]$*. Then the Steiner system* $\mathfrak{S}(9, 3, 2)$ *is*

given as

$$\mathfrak{S} = \{\{1,2,3\},\{4,5,6\},\{7,8,9\},\{1,4,7\},\{2,5,8\},\{3,6,9\},\{1,5,9\},$$
$$\{2,6,7\},\{3,4,8\},\{3,5,7\},\{1,6,8\},\{2,4,9\}\}.$$

The corresponding disjunct matrix is given below.

$$M = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

In [26], it is proved that $\mathfrak{S}(n,b,p)$ Steiner system gives a $\lfloor \frac{b-1}{p-1} \rfloor$-disjunt matrix. A relation between the Steiner system and constant column weight t-disjunct matrices was conjectured, as stated below.

Conjecture 2.27. *[26] Let M be a $n \times m$ t-disjunct matrix with constant column weight b. Let $p = \frac{b+t-1}{t}$ (we assume that p is an integer). The maximum $\mathfrak{S}(n,b,p)$ Steiner system gives a matrix M' that is no worse than M. In other words, $|\mathfrak{S}(n,b,p)| \geq m$.*

Chapter 3

Distributed Multi-User Secret Sharing with Multiple Secrets per User

3.1 Introduction

We consider a distributed multi-user secret sharing (DMUSS) setting in which there is a dealer, n storage nodes, and m secrets. Each user demands a t-subset of m secrets. Earlier work in this setting dealt with the case of $t = 1$ [1]; in this work, we consider general t. The user downloads shares from the storage nodes based on the designed storage structure and reconstructs its secrets. We identify a necessary condition on the storage structures to ensure weak secrecy. We also make a connection between storage structures for this problem and t-disjunct matrices. We apply various t-disjunct matrix constructions in this setting and compare their performance in terms of the number of storage nodes and communication complexity. Finally, we characterize the capacity region of the DMUSS problem when the storage structure is specified.

3.2 System Model

We consider a distributed secret sharing system that comprises n storage nodes, m ($m \geq n$) secrets, and $P = \binom{m}{t}$ users (Fig. 3.1), with the primary goal of enabling the dealer to convey a specific set of secrets to each user securely via storage nodes. In this system model:

- Each secret s_j has a *storage set* $A_j \subseteq [n]$, which represents the set of all storage nodes that store the shares corresponding to secret s_j. For each $i \in A_j$, a share corresponding to secret s_j is stored in i-th storage node.
- Storage nodes are passive, which means they do not communicate with each other. The users do not communicate with each other either.
- Each user u requests a t-subset S_u of $[m]$ secrets, where $|S_u| = t$.
- The dealer has access to all storage nodes but has no access to the users.

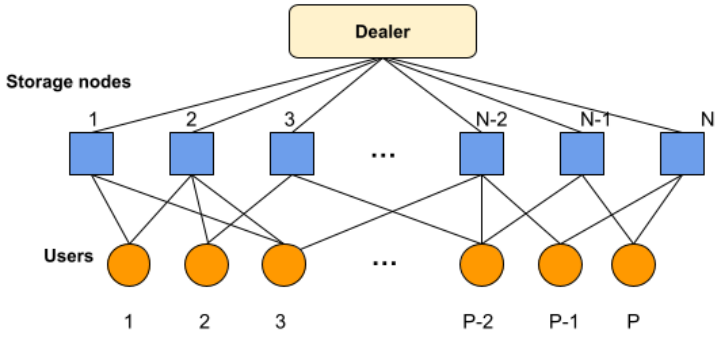

Figure 3.1: System Model

Additionally, we make the following assumption about the secrets.

Assumption 3.1. *All the secrets $s_1, \ldots, s_m$ are uniformly distributed and mutually independent.*

The aim is to develop a distributed secret sharing protocol that encodes secrets into shares and distributes them among storage nodes so that each user u can successfully reconstruct their designated set of t secrets and the secrecy condition is satisfied in a weak sense as defined below.

Definition 3.2. *A distributed secret sharing protocol (DSSP) is a bundle of ($\mathcal{A}$, $\mathcal{E}$, $\boldsymbol{Z}_{n\times h}$, $\mathcal{D}$), where*

- The storage set A_i corresponding to the secret s_i is the only authorized set for this secret. Consequently, the set of all storage sets is the *storage structure* $\mathcal{A}$, i.e.,

$$\mathcal{A} \triangleq \{A_j : j \in [m]\}. \tag{3.1}$$

- The function $\mathcal{E} : \mathbb{F}_q^m \to \mathbb{F}_q^h$ with $h \geq m$ is an encoding function which relates to storage overhead of the system. The input $\boldsymbol{s} = (s_1, s_2, \ldots, s_m)^T$ is a vector of all secrets. The output $\mathbf{y} = \mathcal{E}(\boldsymbol{s}) = (y_1, \ldots, y_h)^T$ is a vector of all data (shares) to be distributed and stored in the storage nodes.

- $\boldsymbol{Z} = [z_{i,r}]_{n\times h}$, where $z_{i,r} = 1$ if y_r is stored in i^{th}-storage node, otherwise 0. We denote by $\mathbf{y}_j$ the vector of all shares stored in nodes indexed by the elements of storage set A_j. The matrix $\boldsymbol{Z}$ is referred to as *storing matrix*. The mapping of the output symbols to the storage nodes (specifying which output symbols are stored at each storage node) is referred to as the *storage profile*.

- (*Correctness* Condition) $\mathcal{D}$ is a collection of m decoding functions $\mathcal{D}_j : \mathbb{F}_q^{|\mathbf{y}_j|} \to \mathbb{F}_q$, such that $\mathcal{D}_j(\mathbf{y}_j) = s_j$. In other words, each user can successfully reconstruct its secrets.

The above definition is the same as the one defined in [1]. Unlike the single notion of secrecy condition for a secret sharing scheme in Definition 2.1, two secrecy conditions can be considered for a DSSP as given below.

Definition 3.3. *A DSSP is weakly secure if a user does not get any information, in an information-theoretic sense, about the individual secrets of any other user. Let U_j denote the set of all the data the user j has access to, $I_j \subseteq [m]$ be the indices of secrets requested by user j. Then*

$$\forall j \in \left[\binom{m}{t}\right],\ \ell \in [m] \setminus I_j :\ H(s_\ell|U_j) = H(s_\ell). \tag{3.2}$$

Definition 3.4. *A DSSP is perfectly secure if no information about all other secrets is revealed to the user j. Let I_j and U_j be as defined above. Then*

$$\forall j \in \left[\binom{m}{t}\right],\ H(s_{I_j^c}|U_j) = H(s_{I_j^c}),\ \textit{where } I_j^c = [m] \setminus I_j. \tag{3.3}$$

The notions of *storage overhead* and *communication complexity* as defined in [1] are recalled below. These are used throughout the paper to evaluate the efficiency of proposed DSSPs.

Remark 3.5. *The total number of $\mathbb{F}_q$-symbols stored in storage nodes is $k' = \sum_{i=1}^{n}\sum_{r=1}^{h} z_{i,r}$, where $\boldsymbol{Z} = [z_{i,r}]_{n\times h}$ is specified in Definition 3.2.*

Definition 3.6. *The storage overhead, SO, of the DSSP is defined as*

$$SO \triangleq \frac{k'}{m}. \tag{3.4}$$

Note that the correctness condition must be satisfied for m uniformly distributed and mutually independent secrets. Therefore, $k' \geq m$ and, consequently, $SO \geq 1$.

Definition 3.7. *Let c_u denote the number of symbols user u needs to download from the storage nodes to reconstruct the designated set of secrets S_u. Then the communication complexity C is defined as*

$$C \triangleq \sum_{u=1}^{\binom{m}{t}} c_u. \tag{3.5}$$

3.3 Weakly Secure DSSP with Optimal Storage Overhead

In this section, we give a necessary condition on storage sets in a DSSP with weak secrecy, which relates to the disjunct matrices used in the *group testing* (refer Section 2.4). We then propose a scheme for constructing DSSPs with optimal storage overhead using the storage structure obtained from disjunct matrices.

3.3.1 Conditions on the Storage Structure

Lemma 3.8. *For any weakly secure DSSP with a storage structure $\mathcal{A}$ defined in* (3.1)*, we have*

$$A_{j_{t+1}} \nsubseteq (A_{j_1} \cup A_{j_2} \cup \cdots \cup A_{j_t}),$$

$\forall\, j_1, j_2, \ldots, j_{t+1} \in [m]$ *with* $j_1 \neq j_2 \neq \cdots \neq j_{t+1}$.

Proof. Assume to the contrary that $A_{j_{t+1}} \subseteq (A_{j_1} \cup A_{j_2} \cup \cdots \cup A_{j_t})$ for some $j_1 \neq j_2 \neq \cdots \neq j_{t+1}$. This means that the user who has access to the secrets $s_{j_1}, \ldots, s_{j_t}$ also has access to the secret $s_{j_{t+1}}$. This implies that the weak secrecy condition in (3.2) is violated, which is a contradiction. □

The collection of subsets satisfying the Lemma 3.8 can be related to the columns of *disjunct* matrices (see Definition 2.21).

Theorem 3.9. *Consider an $n \times m$ matrix M where the columns correspond to the secrets, the rows correspond to the storage nodes, and the support of each column corresponds to the storage set of each secret. Then for a weakly secure DSSP, M is t-disjunct.*

Next, we present a DSSP that generalizes the optimal storage overhead DSSP in [1] to the case where each user requests $t(\geq 1)$ secrets.

3.3.2 DSSP with Optimal Storage Overhead

Consider a system with m secrets, and each user wants to access a subset of t secrets, $t < m$. There can be at most $\binom{m}{t}$ users in the system. Let $\mathcal{A} = \{A_i : i \in [m]\}$ be the storage structure which consists of m subsets, each corresponding to the storage sets of m secrets. Suppose a user requests $\mathcal{P} \subset [m]$ secrets, then the user is given access to all the nodes in $\bigcup_{i \in \mathcal{P}} A_i$. To ensure weak secrecy, the storage structure $\mathcal{A}$ must satisfy the condition specified in Lemma 3.8. So, we consider $\mathcal{A}$ to be a set of supports of each column of a t-disjunct matrix. We consider t-disjunct matrices to have the same column weight (i.e., the same number of non-zero positions in each column). Therefore each storage set is of the same size. Let $|A_i| = r$ and $A_i = \{n_{i,1}, n_{i,2}, \ldots, n_{i,r}\}, \forall\, i \in [m]$.

Remark 3.10. *As each storage node is present in at least one of the storage sets in the storage structure considered, a set of n such secrets always exists. Otherwise, we can ensure weak secrecy using a lesser number of storage nodes. WLOG, let these n secrets be indexed by $[n]$. Thus, we have $\bigcup_{i=1}^{n} A_i = [n]$. This choice of n secrets helps in constructing a one-to-one mapping between these secrets and their corresponding shares.*

Example 3.11.

Algorithm 2 Proposed DSSP

(a) **(Initialization)** Pick n secrets such that the union of their storage sets is $[n]$ (see Remark 3.10).

(b) **(Share Distribution for n secrets)** For each $i \in [n]$, use a variation of (r, r)-Shamir's encoder (see Lemma 3.12) to encode secret s_i into shares $(y_{n_{i,1}}, \ldots, y_{n_{i,r}})$. Each share y_i is stored in i-th storage node, where, by construction, we have $\bigcup_{i\in[n]}\{y_{n_{i,1}}, \ldots, y_{n_{i,r}}\} = \{y_1, \ldots, y_n\}$.

(c) **(Share Distribution for remaining $m - n$ secrets)** For each $i \in [n+1 : m]$, pick distinct, non-zero $\gamma_1, \ldots, \gamma_r \in \mathbb{F}_q$ which are made public. Find a polynomial $P_i(x) = s_i + \sum_{j=1}^{r-1} p_{i,j}x^j$ satisfying $P_i(\gamma_j) = y_{n_{i,j}}$ for each $j \in [r-1]$. The share for secret s_i is $y_i = P_i(\gamma_r)$ which is stored in $n_{i,r}$-th storage node.

Lemma 3.12. *Suppose $q > r$. There is a one-to-one mapping between $(s_1, \ldots, s_n)$ and the shares $(y_1, \ldots, y_n)$ generated in step (b) of the proposed DSSP (Algorithm 2).*

Proof. From step (b) of Algorithm 2, we can construct the following nr linear equations:

$P_1(\gamma_1) = \alpha_{n_{1,1}} y_{n_{1,1}}$	$P_2(\gamma_1) = \alpha_{n_{2,1}} y_{n_{2,1}}$	$\cdots$	$P_n(\gamma_1) = \alpha_{n_{n,1}} y_{n_{n,1}}$
$P_1(\gamma_2) = \alpha_{n_{1,2}} y_{n_{1,2}}$	$P_2(\gamma_2) = \alpha_{n_{2,2}} y_{n_{2,2}}$	$\cdots$	$P_n(\gamma_2) = \alpha_{n_{n,2}} y_{n_{n,2}}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$P_1(\gamma_r) = \alpha_{n_{1,r}} y_{n_{1,r}}$	$P_2(\gamma_r) = \alpha_{n_{2,r}} y_{n_{2,r}}$	$\cdots$	$P_n(\gamma_r) = \alpha_{n_{n,r}} y_{n_{n,r}},$

Table 3.1: Linear system of equations

where $P_i(x) = s_i + \sum_{j=1}^{r-1} p_{i,j}x^j, i \in [n]$ and $\gamma_1, \ldots, \gamma_r \in \mathbb{F}_q$ are distinct and non-zero.

Note that the above system has nr variables. This is because there are $(r-1)n$ unknown variables $p_{i,j}$, for $i \in [n]$ and $j \in [r-1]$, together with n unknown variables $y_1, \ldots, y_n$. We rewrite the linear system as

$$\boldsymbol{A}\boldsymbol{b} + \hat{\boldsymbol{s}} = 0, \tag{3.6}$$

where $\hat{\boldsymbol{s}} = (\overbrace{s_1, \ldots, s_1}^{r \text{ times}}, \ldots, \overbrace{s_n, \ldots, s_n}^{r \text{ times}})$ and $\boldsymbol{b} = (p_{1,1}, \ldots, p_{1,r-1}, \ldots, p_{n,1}, \ldots, p_{n,r-1}, y_1, \ldots, y_n)$. The coefficient matrix $\boldsymbol{A}_{nr \times nr}$ is given below

$$A = \left[\begin{array}{cccc|c} V & & & & T_1 \\ & V & & & T_2 \\ & & \ddots & & \vdots \\ & & & V & T_n \end{array}\right], V = \begin{bmatrix} \gamma_1 & \gamma_1^2 & \cdots & \gamma_1^{r-1} \\ \gamma_2 & \gamma_2^2 & \cdots & \gamma_2^{r-1} \\ \vdots & \vdots & \ddots & \vdots \\ \gamma_r & \gamma_r^2 & \cdots & \gamma_r^{r-1} \end{bmatrix} \in \mathbb{F}_q^{r \times r-1}$$

and all entries in $T_i \in \mathbb{F}_q^{r \times n}$ are zero except $T_i(j, n_{i,j}) = -\alpha_{n_{i,j}}, n_{i,j} \in A_i, j \in [r]$. For the linear system to be feasible, the coefficient matrix $\boldsymbol{A}$ must be invertible. Using [13, Lemma 1], there exist some $\alpha_{n_{i,j}}, \gamma_j \in \mathbb{F}_q \setminus \{0\}$ for $i \in [n], j \in [r]$ such that $\boldsymbol{A}$ is invertible.

Given a non-singular coefficient matrix $\boldsymbol{A}$, any vector of secrets s_i's has a unique solution for y_i's. Hence, there exists a one-to-one mapping between $(s_1, \ldots, s_n)$ and $(y_1, \ldots, y_n)$. □

Lemma 3.13. *Given the shares $y_1, \ldots, y_n$, there is a one-to-one mapping between $(s_{n+1}, \ldots, s_m)$ and the shares $(y_{n+1}, \ldots, y_m)$ generated in step (c) of the proposed DSSP (Algorithm 2).*

Proof. For all $i \in [n+1 : m]$ to find the polynomial P_i, we need to solve the following linear system:

$$\begin{bmatrix} \gamma_1 & \gamma_1^2 & \cdots & \gamma_1^{r-1} \\ \gamma_2 & \gamma_2^2 & \cdots & \gamma_2^{r-1} \\ \vdots & \vdots & \ddots & \vdots \\ \gamma_{r-1} & \gamma_{r-1}^2 & \cdots & \gamma_{r-1}^{r-1} \end{bmatrix} \begin{bmatrix} p_{i,1} \\ p_{i,2} \\ \vdots \\ p_{i,r-1} \end{bmatrix} = \begin{bmatrix} y_{n_{i,1}} - s_i \\ y_{n_{i,2}} - s_i \\ \vdots \\ y_{n_{i,r-1}} - s_i \end{bmatrix}. \tag{3.7}$$

The left-hand side matrix is the Vandermonde matrix which is full rank if and only if γ_j's are all distinct. Consequently, there exists a unique polynomial P_i satisfying the linear system. The share y_i is computed as $P_i(\gamma_r)$, i.e.,

$$y_i = s_i + \sum_{j=1}^{r-1} p_{i,j} \gamma_r^j,$$

where the summation is a constant given $y_1, \ldots, y_n$. This is indeed the required one-to-one mapping between y_i and s_i. □

Using this result, we show that the proposed DSSP is weakly secure, achieving the optimal storage overhead.

Theorem 3.14. *The proposed protocol (Algorithm 2) is a weakly secure DSSP satisfying all the conditions in Definition 3.2. Additionally, it achieves the optimal storage overhead equal to one.*

Proof. **(Correctess)** In this protocol, each user j has access to all the $\sum_{i\in I_j}|A_i|$ evaluations of the polynomials associated with the secrets (with indices I_j) the user has requested. Hence, the correctness condition is satisfied by invoking Shamir's decoder.

(Weak secrecy) Now, we show that the proposed DSSP is indeed a weakly secure DSSP by showing that the condition specified in (3.2) holds. First, we show that the data symbols $y_1,\dots,y_m$ generated according to the proposed protocol are uniformly distributed and mutually independent. As the vector of all secrets is assumed to be full entropy, $(s_1,\dots,s_n)$ is also full entropy. Also by Lemma 3.12, there is a one-on-one relationship between $(s_1,\dots,s_n)$ and $(y_1,\dots,y_n)$. This implies $(y_1,\dots,y_n)$ is also full entropy. Then,

$$\begin{aligned} H(y_{n+1},\dots,y_m|y_1,\dots,y_n) &= H(s_{n+1},\dots,s_m|y_1,\dots,y_n) && \text{(by Lemma 3.13)}\\ &\overset{(*)}{=} H(s_{n+1},\dots,s_m)\\ &= (m-n)\log q && \text{(by Assumption 3.1)} \end{aligned}$$

where $(*)$ holds since $y_1,\dots,y_n$ is independent of $s_{n+1},\dots,s_m$. Using this together with the chain rule, we have

$$\begin{aligned} H(y_1,\dots,y_m) &= H(y_1,\dots,y_n) + H(y_{n+1},\dots,y_m|y_1,\dots,y_n)\\ &= n\log q + (m-n)\log q\\ &= m\log q. \end{aligned} \tag{3.8}$$

Hence, from (3.8), we can say that the shares have full entropy and are mutually independent. As the storage sets are assumed to satisfy the t-disjunctness condition, there exists at least one γ_k, $k\in[r]$ such that $P_i(\gamma_k)$ for some $i\in[m]$ is not accessed by user j. Let this data symbol $P_i(\gamma_k)$ be denoted by $y_i^{(-j)}$. Then for each user $j\in\left[\binom{m}{t}\right]$, and secret $i\in[m]\setminus I_j$, we have

$$\begin{aligned} H(s_i|U_j) &\geq H(s_i|\mathbf{y}\setminus y_i^{(-j)}) && \text{(Conditioning reduces entropy)}\\ &\overset{(\dagger)}{=} H(y_i^{(-j)}|\mathbf{y}\setminus y_i^{(-j)})\\ &= H(y_i^{(-j)}) && \text{(since } y_i^{(-j)} \perp\!\!\!\perp \mathbf{y}\setminus y_i^{(-j)})\\ &= \log q, \end{aligned} \tag{3.9}$$

where $(\dagger)$ holds because given any $r-1$ evaluations of P_i which is the evaluation polynomial corresponding to the user i, out of r available ones, there is a one-to-one mapping between the remaining

evaluation of P_i and s_i and the last equality holds by noting that data shares are all full entropy. Also, for each user $j \in \left[\binom{m}{t}\right]$, and secret $i \in [m] \setminus I_j$ we have

$$H(s_i|U_j) \leq H(s_i) = \log q. \tag{3.10}$$

From (3.9) and (3.10), the weak secrecy condition (3.2) is satisfied.

The storage overhead of the proposed protocol is given by:

$$\text{Storage overhead} = \frac{\text{Total Number of shares}}{\text{Total Number of secrets}} = \frac{n + (m-n)}{m} = 1.$$

□

Next, we demonstrate the algorithm through an example.

Example 3.15. *Let $m = 12$, $n = 9$, and the storage structure be given by the Steiner system from Example 2.26, i.e.,*

$$\mathcal{A} = \{\{1,2,3\},\{4,5,6\},\{7,8,9\},\{1,4,7\},\{2,5,8\},\{3,6,9\},\{1,5,9\},$$
$$\{2,6,7\},\{3,4,8\},\{3,5,7\},\{1,6,8\},\{2,4,9\}\}.$$

The dealer wants to securely share $m = 12$ secrets $s_1, \ldots, s_{12}$, whose storage sets are given by $\mathcal{A}$. At first, the dealer picks the first $n = 9$ secrets $s_1, \ldots, s_9$ according to step (a) of Algorithm 2.

To encode these secrets, according to step (b) of Algorithm 2, the dealer solves (3.6) *with $q = 5$, and the evaluation points $\gamma_1 = 1, \gamma_2 = 2, \gamma_3 = 3$. The shares obtained are given by the following equations:*

$$y_1 = 2s_1 + 3s_2 + 2s_3 + 3s_4 + 4s_5 + 4s_7 + 3s_8 + 2s_9$$
$$y_2 = 2s_1 + s_2 + 3s_3 + 2s_5 + s_7 + s_8 + 2s_9$$
$$y_3 = s_1 + 2s_2 + 4s_4 + 3s_5 + 2s_8 + 2s_9$$
$$y_4 = s_1 + 2s_3 + s_4 + 3s_5 + s_6 + s_7 + 2s_8 + 3s_9$$
$$y_5 = 4s_1 + s_2 + 2s_3 + 4s_5 + 4s_6 + 2s_7 + 4s_8 + 2s_9$$
$$y_6 = 4s_1 + 3s_2 + 4s_3 + 3s_4 + 4s_6 + s_7 + 2s_8 + 2s_9$$
$$y_7 = 2s_1 + 2s_2 + s_3 + 3s_4 + 3s_5 + 4s_6$$
$$y_8 = s_1 + s_2 + s_3 + 3s_5 + 3s_6 + 3s_7 + 2s_8 + 2s_9$$
$$y_9 = 4s_3 + 2s_4 + 3s_5 + 2s_6 + s_7 + 4s_8 + 4s_9.$$

The remaining secrets are encoded according to step (c) of Algorithm 2, taking the same evaluation points as above. The corresponding shares are computed as

$$y_{10} = s_{10} + 2y_3 + 3y_5$$
$$y_{11} = s_{11} + 2y_1 + 3y_6$$
$$y_{12} = s_{12} + 2y_2 + 3y_4.$$

The shares, together with the storage profile, are shown in the following table.

Node 1	Node 2	Node 3	Node 4	Node 5	Node 6	Node 7	Node 8	Node 9
y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9
						y_{10}	y_{11}	y_{12}

3.4 Comparison between Different Constructions of Disjunct Matrices

We compare the number of storage nodes n and communication complexity C when different constructions introduced in Section 2.4 are used as the storage structure in the DSSP described in Section 3.3.

For a given total number of secrets m and the number of secrets t each user requests ($t << m$), we compare the number of storage nodes n and the communication complexity C across the constructions in Section 2.4 for t-disjunct matrices (see also Table 3.2).

Kautz-Singleton (KS) Vs Porat-Rothschild (PR)

There are two regimes considered in the literature: (i) $t = O(poly(\log m))$, and (ii) $t = O(m^{\alpha}),\ \alpha \in (0, 1/2)$. In regime (i), we have $n_{PR} < n_{KS}$ and $C_{PR} < C_{KS}$, which shows that PR is better than KS. Whereas in regime (ii), both the inequalities are reversed, which shows that KS is better than PR.

Kautz-Singleton (KS) Vs Sparse Disjunct ($\ell = 1$) (SD)

Since we have $t << m$, $n_{KS} < n_{SD}$ and $C_{KS} > C_{SD}$. There is a tradeoff between these storage structures, so if, for example, one seeks to minimize the number of storage nodes, KS is better than SD while paying for a higher communication complexity and vice versa.

Porat-Rothschild (PR) vs Sparse disjunct ($\ell = 1$) (SD)

Since we have $t << m$, $n_{PR} < n_{SD}$ and $C_{PR} < C_{SD}$. Thus, PR is better than SD.

	Kautz-Singleton	Porat-Rothschild	Sparse disjunct ($\ell = 1$)
Code	$\mathcal{C}_{out} : [q_1, k, q_1 - k + 1]_{q_1}$- RS code $\mathcal{C}_{in} : I_{q_1}$	$\mathcal{C}_{out} : [\mathscr{z}, k, \delta\mathscr{z}]_{q_1}$-Linear code where $\mathscr{z} \leq \frac{k}{1-H_{q_1}(\delta)}$ $\mathcal{C}_{in} : I_{q_1}$	$\mathcal{C}_{out} : [t+1, k=2]_{q_2}$-RS code where $q_2 = \sqrt{m}$ $\mathcal{C}_{in} : I_{q_2}$
Disjunctness	$t = \lfloor \frac{q_1 - 1}{k-1} \rfloor$	$t + 1 = \lceil \frac{1}{1-\delta} \rceil$	$t + 1 \leq \sqrt{m}$
t-disjunct Matrix	$\mathcal{C}^* : q_1^2 \times m$ $\approx t^2 \log_t^2 m \times m$	$\mathcal{C}^* : \mathscr{z}q_1 \times m$ $\approx t^2 \log m \times m$	$\mathcal{C}^* : (t+1)q_2 \times m$ $= (t+1)\sqrt{m} \times m$
Column weight	$q_1 \approx t \log_t m$	$\mathscr{z}$	$t + 1$
Storage Overhead	1	1	1
Comm. Complexity	$\binom{m-1}{t-1} \times q_1 \times m$	$\binom{m-1}{t-1} \times \mathscr{z} \times m$	$\binom{m-1}{t-1} \times (t+1) \times m$

Table 3.2: Comparison of disjunct matrix constructions.

Remark 3.16. *If Conjecture 2.27 is true, then for a given n, the number of storage nodes, and b, the size of storage sets, the storage structure obtained from Steiner system $\mathfrak{S}(n, b, p)$ is the best in terms of accommodating more number of secrets.*

We prove this conjecture for the special cases where the matrix M is obtained using the Kautz-Singleton and Sparse Disjunct constructions.

Consider a $q^2 \times q^k$ t-disjunct matrix obtained from Kautz-Singleton construction, $t = \lfloor \frac{q-1}{k-1} \rfloor$. We set $k = \frac{q+t-1}{t}$ to accommodate most secrets in this construction. Then the corresponding Steiner system with the same number of storage nodes is given by $\mathfrak{S}(q^2, q, k)$. The following lemma compares the total number of secrets accommodated by both constructions.

Lemma 3.17. *If the Steiner system $\mathfrak{S}(q^2, q, p = \frac{q+t-1}{t})$ exists, then it can accommodate more secrets compared to the storage structure obtained from the Kautz-Singleton construction with the same number of nodes q^2 and constant column weight q. In other words,*

$$q^s \leq \left|\mathfrak{S}(q^2, q, p)\right| = \binom{q^2}{s} \Big/ \binom{q}{s}. \tag{3.11}$$

Proof. Expanding the binomial coefficients on RHS of (3.11) and observing that for all $\ell \in [1, q)$, $\frac{q^2-\ell}{q-\ell} > q$ gives (3.11). □

Similarly, a comparison between a sparse disjunct matrix and the Steiner system is given below.

Lemma 3.18. *If the Steiner system $\mathfrak{S}((t+1)q, t+1, 2)$ exists, then it can accommodate more secrets compared to the storage structure obtained from the Sparse Disjunct matrix with the same number of*

nodes $(t+1)q$ *and constant column weight* $t+1$. *In other words,*

$$q^2 \leq |\mathfrak{S}((t+1)q, t+1, 2)| = \binom{(t+1)q}{2} \Big/ \binom{t+1}{2}.$$

Balanced Storage Profile

In large-scale distributed storage systems, it is essential to distribute the data evenly across the nodes and ensure each node has a similar amount of data to manage. This helps in avoiding problems like slower access times and system failures. We say that a collection $\mathcal{F}$ of subsets of $[n]$ is a *balanced collection* if each $i \in [n]$ belongs to the same number of subsets in $\mathcal{F}$. The Kautz-Singleton construction and the Sparse disjunct matrices defined above provide a t-disjunct matrix with constant row and column weights. Thus, the resulting storage structures are balanced collections and can be used to obtain a DSSP with a balanced storage profile.

3.5 Capacity Region of Distributed Multi-User Secret Sharing

In [13], the capacity region of the distributed multi-user secret sharing system is characterized, as subject to correctness and secrecy constraints. They consider a DMUSS system that consists of a dealer, $n \in \mathbb{N}$ storage nodes, and $m \in \mathbb{N}$ users. In the system set-up considered in [1], the storage structure has a regularized form, and the user's secret messages have equal size, whereas the DMUSS set-up in [13] considers an arbitrary storage structure, and the users can have different secret sizes. Now, we recall a few definitions given in [13]:

Definition 3.19. *Let the length of secret* s_i *be denoted by* r_i*; we define its rate as* $R_i = \frac{r_i}{K}$*, where* K *is the size of each storage node. Let secret* $s_i = (s_{i,0}, \ldots, s_{i,R_i-1})$*, where* $s_{i,j} \in \mathbb{F}_q^K$.

Definition 3.20. *The capacity region of a DMUSS is defined as a set of all achievable rate tuples, subject to the correctness and secrecy constraints.*

The capacity region of DMUSS, characterized in [13], is a special case of Theorem 3.21 where $t = 1$. We generalize the techniques to deduce the following result.

Theorem 3.21. *The capacity region of DMUSS is the convex hull of all regions with the rate tuple* $(R_1, \ldots, R_m)$ *satisfying:*

$$R_i \leq \min |A_i \setminus \cup_{j \in S} A_j|, \ \forall\, S \subseteq [m] \setminus \{i\}, |S| = t \tag{3.12}$$

$$\sum_{i \in S} R_i \leq |\cup_{i \in S} A_i|, \quad S \subseteq [m]. \tag{3.13}$$

Proof Sketch. To prove this theorem, we need to verify the following two statements:

- **Achievability.** For every rate tuple $(R_1, \ldots, R_m) \in (\mathbb{N} \cup \{0\})^m$ satisfying (3.12) and (3.13), there exists a DSSP that is weakly secure according to Definition 3.3. The convexity property then follows from [13, Lemma 4].
- **Converse.** For every sequence of weakly secure DSSP, the rate tuple $(R_1, R_2, \ldots, R_m)$ satisfy (3.12) and (3.13).

Remark 3.22. *Suppose* $(R_1, \ldots, R_m)$ *satisfy* (3.12) *and* (3.13). *Then by convexity of the capacity region, there exists* $(R'_1, \ldots, R'_m) \in (\mathbb{N} \cup \{0\})^m$ *such that* $\sum_{i=1}^m R'_i = n$ *and*

$$\sum_{i \in S} R_i \leq \sum_{i \in S} R'_i \leq |\cup_{i \in S} A_i|, \ \forall\, S \subseteq [m].$$

Algorithm 3 Variable Secret Size DSSP

(a) **(Initialization)** Pick tuples $(R'_1, \ldots, R'_m) \in (\mathbb{N} \cup \{0\})^m$ as mentioned in Remark 3.22

(b) **(Share Distribution)** For each $i \in [m]$, suppose $A_i = \{n_{i,1}, \ldots, n_{i,|A_i|}\}$. Pick distinct, non-zero $\gamma_{i,1}, \ldots, \gamma_{i,|A_i|} \in \mathbb{F}_q$ which are made public. Solve the following equations

$$s_{i,0} + \cdots + s_{i,R_i-1}\gamma_{i,j}^{R_i-1} + p_{i,R_i}\gamma_{i,j}^{R_i} + \cdots + p_{i,|A_i|-1}\gamma_{i,j}^{|A_i|-1} = y_{n_{i,j}}, \ \forall j \in [|A_i|] \tag{3.14}$$

where the secret $s_i = (s_{i,0}, \ldots, s_{i,R_i-1})$ and $p_{i,R_i}, \ldots, p_{i,R'_i-1}$ (chosen randomly) are known and $p_{i,R'_i}, \ldots, p_{i,|A_i|-1}$ and $y_{n_{i,1}}, \ldots, y_{n_{i,|A_i|}}$ are unknown. Store $y_{n_{i,j}}$ in $n_{i,j}$-th storage node for each $j \in [|A_i|]$.

Lemma 3.23. *There is a unique solution to the linear system* (3.14).

Proof. We rewrite the equations as

$$\sum_{k=R'_i}^{|A_i|-1} p_{i,k}\gamma_{i,j}^k - y_{n_{i,j}} = -\sum_{k=0}^{R_i-1} s_{i,k}\gamma_{i,j}^k - \sum_{k=R_i}^{R'_i-1} p_{i,k}\gamma_{i,j}^k \equiv \kappa_{i,j}, \ \forall j \in [|A_i|], i \in [m]$$

such that the unknowns are on the left-hand side and known variables are on the right-hand side. By vectorizing these equations, we get

$$\boldsymbol{A}\boldsymbol{b} + \boldsymbol{s} = \boldsymbol{0},$$

where $\boldsymbol{b} = (p_{1,R'_1}, \ldots, p_{1,|A_1|-1}, \ldots, p_{m,R'_m}, \ldots, p_{m,|A_m|-1}, \boldsymbol{y})$, $\boldsymbol{y} = (y_1, \ldots, y_n)$ since $\cup_{i\in m} A_i = [n]$ and $\boldsymbol{s} = (\kappa_{1,1}, \ldots, \kappa_{1,|A_1|}, \ldots, \kappa_{m,1}, \ldots, \kappa_{m,|A_m|})$. The coefficient matrix $\boldsymbol{A}_{N\times N}$, $N = \sum_{i=1}^{m} |A_i|$, is given below

$$A = \left[\begin{array}{cccc|c} V_1 & & & & T_1 \\ & V_2 & & & T_2 \\ & & \ddots & & \vdots \\ & & & V_m & T_m \end{array}\right], V_i = \begin{bmatrix} \gamma_{i,1}^{R'_i} & \gamma_{i,1}^{R'_i+1} & \cdots & \gamma_{i,1}^{|A_i|-1} \\ \gamma_{i,2}^{R'_i} & \gamma_{i,2}^{R'_i+1} & \cdots & \gamma_{i,2}^{|A_i|-1} \\ \vdots & \vdots & \ddots & \vdots \\ \gamma_{i,|A_i|}^{R'_i} & \gamma_{i,|A_i|}^{R'_i+1} & \cdots & \gamma_{i,|A_i|}^{|A_i|-1} \end{bmatrix}$$

where all entries in T_i are zero except $T_i(j, n_{i,j}) = -1, n_{i,j} \in A_i, j \in [|A_i|]$. For the linear system to have a unique solution, the coefficient matrix $\boldsymbol{A}$ must be invertible. Once again, using [13, Lemma 1], we see that $\boldsymbol{A}$ is non-singular. □

Proof of Theorem 3.21. **(Achievability)** Suppose that the m secret messages have rates $(R_1, \ldots, R_m)$ that satisfy (3.12) and (3.13). Let the secrets be distributed according to Algorithm 3. The user j requesting t-subset of secrets with indices I_j has access to the shares in $\cup_{i\in I_j} A_i$. For a secret s_i, $i \in I_j$, the user has $|A_i|$ evaluations of $|A_i| - 1$ degree polynomial. By Lemma 3.23, there is a unique solution to the polynomial. Thus, the correctness condition holds.

Define $P = \bigcup_{i\in[m]} \{p_{i,R_i}, \ldots, p_{i,|A_i|-1}\}$ and $O = \bigcup_{i\in[m]} \{p_{i,R_i}, \ldots, p_{i,R'_i-1}\}$. Since $R'_i \le |A_i|$ for all $i \in [m]$, we have $O \subseteq P$. Consider the joint entropy of the shares:

$$\begin{aligned} H(y_1, \ldots, y_n) &\overset{(a)}{=} H(y_1, \ldots, y_n) + H(s_1, \ldots, s_m, P | y_1, \ldots, y_n) \\ &\ge H(y_1, \ldots, y_n) + H(s_1, \ldots, s_m, O | y_1, \ldots, y_n) && \text{(since } O \subseteq P\text{)} \\ &= H(s_1, \ldots, s_m, y_1, \ldots, y_n, O) && \text{(Chain rule of entropy)} \\ &\ge H(s_1, \ldots, s_m, O) && \text{(since uncertainty reduces)} \\ &= \sum_{i=1}^{m} H(s_i) + H(O) && (s_1 \perp\!\!\!\perp \cdots \perp\!\!\!\perp s_m \perp\!\!\!\perp O) \\ &= \sum_{i=1}^{m} R'_i \log q \\ &= n \log q && \text{(by Remark 3.22)} \end{aligned}$$

where (a) holds since the shares uniquely determine all the coefficients of the polynomial, which are s_i's and $p_{i,j}$'s. On the other hand, $H(y_1, \ldots, y_n) \le n \log q$. Therefore, the shares have full entropy.

Consider two users, i and j. The rate for the user i satisfies $R_i \leq |A_i \setminus \cup_{k\in I_j} A_k|$. Let $A_i^{(-j)}$ be an arbitrary subset of $A_i \setminus \cup_{k\in I_j} A_k$ with size R_i and $\mathbf{y}_i^{(-j)}$ be the set of shares stored in the corresponding storage nodes. Let $\boldsymbol{p}_i = (p_{i,R_i+1}, \ldots, p_{i,|A_i|-1})$. Then

$$
\begin{aligned}
H(s_i|U_j) &\geq H(s_i|\mathbf{y} \setminus \mathbf{y}_i^{(-j)}) && \text{(Conditioning reduces entropy)} \\
&\overset{(*)}{=} H(s_i|\mathbf{y} \setminus \mathbf{y}_i^{(-j)}) + H(\boldsymbol{p}_i|\mathbf{y} \setminus \mathbf{y}_i^{(-j)}, s_i) + H(\mathbf{y}_i^{(-j)}|\mathbf{y} \setminus \mathbf{y}_i^{(-j)}, s_i, \boldsymbol{p}_i) \\
&= H(\mathbf{y}_i^{(-j)}, s_i, \boldsymbol{p}_i|\mathbf{y} \setminus \mathbf{y}_i^{(-j)}) && \text{(Chain rule of entropy)} \\
&= H(\mathbf{y}_i^{(-j)}|\mathbf{y} \setminus \mathbf{y}_i^{(-j)}) + H(s_i, \boldsymbol{p}_i|\mathbf{y}_i^{(-j)}, \mathbf{y} \setminus \mathbf{y}_i^{(-j)}) && \text{(Chain rule of entropy)} \\
&= H(\mathbf{y}_i^{(-j)}|\mathbf{y} \setminus \mathbf{y}_i^{(-j)}) + H(s_i, \boldsymbol{p}_i|\mathbf{y}) \\
&\geq H(\mathbf{y}_i^{(-j)}|\mathbf{y} \setminus \mathbf{y}_i^{(-j)}) && \text{(since entropy is nonnegative)} \\
&= H(\mathbf{y}_i^{(-j)}) && \text{(since } \mathbf{y}_i^{(-j)} \perp\!\!\!\perp \mathbf{y} \setminus \mathbf{y}_i^{(-j)}\text{)} \\
&= R_i \log q.
\end{aligned}
$$

Consider the following to establish $(*)$. Given $\boldsymbol{p}_i$ and s_i, the shares $\boldsymbol{y}_i$ in A_i can be computed using (3.14) and consequently a subset $\mathbf{y}_i^{(-j)}$. This shows that $H(\mathbf{y}_i^{(-j)}|\mathbf{y} \setminus \mathbf{y}_i^{(-j)}, s_i, \boldsymbol{p}_i) = 0$. Using (3.14) again, given $\mathbf{y} \setminus \mathbf{y}_i^{(-j)}$ (in particular $\mathbf{y}_i \setminus \mathbf{y}_i^{(-j)}$) and s_i, we can form a linear system in $\boldsymbol{p}_i$ with $|A_i| - R_i$ variables. The number of equations is $\mathbf{y}_i \setminus \mathbf{y}_i^{(-j)} = |A_i| - R_i$ which can be solved to find unique $\boldsymbol{p}_i$ (Lemma 3.23). Therefore, $H(\boldsymbol{p}_i|\mathbf{y} \setminus \mathbf{y}_i^{(-j)}, s_i) = 0$. On the other hand,

$$H(s_i|U_j) \leq H(s_i) = R_i \log q. \tag{3.15}$$

Thus, the weak secrecy condition (3.2) is satisfied.

This shows the existence of a weakly secure DSSP for the given values of rate tuples, which completes the achievability proof.

(Converse) Consider two users, i and j. Let $S_i^{(j)}$ denote the secrets that are common to both the users and $S_i^{(-j)}$ be the secrets of user i that are not requested by user j. Notice that the shares of user i can be written as $U_i = \mathbf{y}_{S_i^{(j)}} \cup \mathbf{y}_{S_i^{(-j)}}$. Then by the correctness and weak secrecy conditions, we have

$$H(s_i|\mathbf{y}_{S_i^{(j)}}, \mathbf{y}_{S_i^{(-j)}}) = 0, \quad H(s_i|\mathbf{y}_{S_i^{(j)}}) = H(s_i). \tag{3.16}$$

Consider the following mutual information:

$$
\begin{aligned}
I(\mathbf{y}_{S_i^{(-j)}}; s_i|\mathbf{y}_{S_i^{(j)}}) &= H(\mathbf{y}_{S_i^{(-j)}}|\mathbf{y}_{S_i^{(j)}}) - H(\mathbf{y}_{S_i^{(-j)}}|\mathbf{y}_{S_i^{(j)}}, s_i) && \text{(by definition)} \\
&\leq H(\mathbf{y}_{S_i^{(-j)}}|\mathbf{y}_{S_i^{(j)}}) && \text{(since entropy is nonnegative)} \\
&\leq H(\mathbf{y}_{S_i^{(-j)}}) && \text{(Conditioning reduces entropy)} \\
&\leq |S_i^{(-j)}| \log q \\
&\leq |A_i \setminus \cup_{k\in I_j} A_k| \log q, \\
I(\mathbf{y}_{S_i^{(-j)}}; s_i|\mathbf{y}_{S_i^{(j)}}) &= H(s_i|\mathbf{y}_{S_i^{(j)}}) - H(s_i|\mathbf{y}_{S_i^{(j)}}, \mathbf{y}_{S_i^{(-j)}}) && \text{(by definition)} \\
&= H(s_i) && \text{(by (3.16))} \\
&= R_i \log q, \\
\implies R_i \log q &\leq |A_i \setminus \cup_{k\in I_j} A_k| \log q \\
\implies R_i &\leq |A_i \setminus \cup_{k\in I_j} A_k|.
\end{aligned}
$$

This establishes the condition in (3.12).

Consider a set $S \subseteq [m]$, where $S = \{j_1, \ldots, j_k\}$. Then by correctness condition, we have

$$
H(s_{j_1}, \ldots, s_{j_k}|\mathbf{y}_{j_1}, \ldots, \mathbf{y}_{j_k}) = 0, \quad H(s_{j_1}, \ldots, s_{j_k}) = \sum_{j\in S} R_j \log q. \tag{3.17}
$$

Consider the following mutual information:

$$
\begin{aligned}
I(\mathbf{y}_{j_1}, \ldots, \mathbf{y}_{j_k}; s_{j_1}, \ldots, s_{j_k}) &= H(\mathbf{y}_{j_1}, \ldots, \mathbf{y}_{j_k}) - H(\mathbf{y}_{j_1}, \ldots, \mathbf{y}_{j_k}|s_{j_1}, \ldots, s_{j_k}) && \text{(by definition)} \\
&\leq H(\mathbf{y}_{j_1}, \ldots, \mathbf{y}_{j_k}) \\
&\leq |\cup_{j\in S} A_j| \log q, \\
I(\mathbf{y}_{j_1}, \ldots, \mathbf{y}_{j_k}; s_{j_1}, \ldots, s_{j_k}) &= H(s_{j_1}, \ldots, s_{j_k}) - H(s_{j_1}, \ldots, s_{j_k}|\mathbf{y}_{j_1}, \ldots, \mathbf{y}_{j_k}) && \text{(by definition)} \\
&= \sum_{j\in S} R_j \log q, && \text{(by (3.17))} \\
\implies \sum_{j\in S} R_j \log q &\leq |\cup_{j\in S} A_j| \log q \\
\implies \sum_{j\in S} R_j &\leq |\cup_{j\in S} A_j|.
\end{aligned}
$$

This establishes the condition in (3.13), which completes the converse proof. □

Chapter 4

Decoding Reed-Muller Codes

In this chapter, we introduce Reed-Muller codes and list some useful properties. Then we describe some efficient decoding algorithms which exploit the symmetry from the automorphism group of the code. In particular, we review Recursive Puncturing–Aggregation (RXA) Decoding of $RM(m, r)$ [18] and Maximum Likelihood (ML) Syndrome Decoding of $RM(m, m-3)$ [19]. Then, we propose an RXA-Syndrome Decoding algorithm to decode $RM(m, m-4)$ code.

Definition 4.1. *A boolean function $f : \mathbb{F}_2^m \to \mathbb{F}_2$ is a polynomial in $(x_1, \ldots, x_m) \in \mathbb{F}_2^m$ which is represented as*

$$f(x_1, \ldots, x_m) = \sum_{A \subseteq [m]} c_A \prod_{i \in A} x_i,$$

where the degree of the monomial $\prod_{i \in A} x_i$ is $|A|$. The degree of the polynomial, denoted by $\deg(f)$, is the highest among the degrees of the monomials with non-zero coefficients. Let $i = (i_1, \ldots, i_m)$ be the binary representation of i, and let $v_i(f)$ denote the evaluation of the polynomial at i, i.e., $v_i(f) = f(i_1, \ldots, i_m)$. The evaluation vector $v(f) = (v_1(f), \ldots, v_{2^m}(f))$ is the vector of all 2^m evaluation points.

Definition 4.2. *The r-th order binary Reed-Muller code $RM(m, r)$ of length $n = 2^m$, for $0 \leq r \leq m$, is the set of all vectors $v(f)$, where $f(x_1, \ldots, x_m)$ is a Boolean function which is a polynomial of degree at most r. In other words,*

$$RM(m, r) = \{v(f) : \deg(f) \leq r\}.$$

Proposition 4.3. *The dimension of $RM(m, r)$ code is $k = 1 + \binom{m}{1} + \cdots + \binom{m}{r}$ and minimum distance is $d = 2^{m-r}$. Thus, it is an $[2^m, \sum_{i=0}^{r} \binom{m}{i}, 2^{m-r}]_2$ linear code.*

The following result gives an equivalent recursive definition of RM codes.

Proposition 4.4. $RM(m, r) = \{(u, u+v) : u \in RM(m-1, r), v \in RM(m-1, r-1)\}$.

Using the above definition, it is easy to see that puncturing the first or the second half leaves shorter $RM(m-1,r)$ of the same order, and adding both halves projects onto shorter $RM(m-1,r-1)$ of lower order.

Proposition 4.5. *$RM(m, m-r-1)$ is the dual code to $RM(m,r)$, for $0 \leq r \leq m-1$.*

Consider an affine transform $\pi(A,c)$ on $\mathbb{F}_2^m$ as a map $x = (x_1,\ldots,x_m)^T \mapsto x_\pi = Ax + c$ for some invertible matrix $A \in \mathbb{F}_2^{m\times m}$ and $c \in \mathbb{F}_2^m$. Since the map $y \mapsto A^{-1}(y-c)$ is inverse of π, π is bijective. Thus, $\pi(A,c)$ can be thought of as permuting the coordinates of x. Let $v_\pi(f)$ denote the evaluation vector corresponding to $f \circ \pi$ polynomial. The following result gives a connection between affine transformations and the automorphism group (see Definition 2.15) of RM codes.

Proposition 4.6. *Let $\pi(A,c)$ be an invertible affine transform and $f : \mathbb{F}_2^m \to \mathbb{F}_2$ a boolean function. Then, for all $x = (x_1,\ldots,x_m)$, we have*

$$\deg(f(Ax+c)) \leq \deg(f(x)).$$

Moreover, $v_\pi(f)$ is a permutation of $v(f)$. Then, we get

$$\mathrm{Aut}(RM(m,r)) = \{\pi(A,c) \,:\, A \in \mathbb{F}_2^{m\times m} \text{ is invertible}, c \in \mathbb{F}_2^m\}. \tag{4.1}$$

Identifying the automorphisms of RM codes helps in the design of decoding algorithms. Permuting a codeword in $RM(m,r)$ code results in a different codeword in the same code (from Equation (4.1)). So instead of performing operations like puncturing and projection on a single codeword alone, we can perform them on all the permuted codewords.

Next, we consider the decoding algorithms for RM codes. In Section 4.1, we review the Syndrome Decoding of linear codes. In Section 4.2, we review a maximum likelihood syndrome decoding of $RM(m,m-3)$ [19], which exploits the symmetry of the automorphism group of RM codes to reduce the complexity of Syndrome Decoding. The corresponding algorithm performs hard-decision decoding by considering a binary symmetric channel (BSC) model for noise. We review the Recursive Puncturing–Aggregation (RXA) Decoding algorithm of $RM(m,r)$ in Section 4.3, which performs soft-decision decoding, i.e., it models the noisy channel as an additive white Gaussian noise (AWGN) channel and consequently works with log-likelihood ratios in the received vector. Later in Section 4.4, we combine these two algorithms to propose a hard-decision RXA-Syndrome decoder for $RM(m,m-4)$ codes. As a result, we only consider RXA (Section 4.3) for the BSC model.

4.1 Syndrome Decoding of Linear Codes

Suppose that a codeword $x \in C \subseteq \mathbb{F}_q^n$ is transmitted, and $y \in \mathbb{F}_q^n$ is received such that $y = x + e$ and $e \in \mathbb{F}_q^n$ is the error. If H is the parity check matrix, then $Hy^T = He^T$, since $Hx^T = 0$ for all $x \in C$. Below, we define the notion of syndrome and cosets of C.

Definition 4.7 (Syndrome)**.** *Let $C \subseteq \mathbb{F}_q^n$ be a linear code with parity check matrix H. For a received vector $y \in \mathbb{F}_q^n$, $Hy^T \in \mathbb{F}_q^n$ is called the syndrome of y. The set of all received vectors with the same syndrome is called a coset of the code C. The coset leader $x \in \mathbb{F}_q^n$ is an element of the coset with the least Hamming weight, and we represent each coset as $x + C = \{x + c : c \in C\}$ where x is the coset leader.*

In case of multiple vectors in the coset have the least Hamming weight, we pick one of them at random to be the coset leader. Thus, we can assume that the coset leader x is unique for each coset such that it is represented as $x + C$.

Proposition 4.8. *Let C be a $[n, k]_q$ linear code. Then, there is a one-to-one correspondence between syndrome $s \in \mathbb{F}_q^{n-k}$ and the coset leader. Thus, there are 2^{n-k} cosets, each containing q^k elements (equal to $|C|$).*

The above result suggests a simple decoding algorithm called Syndrome Decoding.

Algorithm 4 Syndrome Decoding

(a) Construct a lookup table containing all possible syndromes and their corresponding coset leaders.

(b) Compute the syndrome Hy^T for the received vector y and lookup in the table for the matching syndrome-coset leader pair.

(c) Output the result as $c = y + e$, where e is the coset leader.

Remark 4.9. *Since $d_H(c, y) = \mathrm{wt}(c+y) = \mathrm{wt}(e) \leq \mathrm{wt}(c'+y) = d_H(c', y)$ for all $c' \in C$. This shows that the output codeword is the closest codeword to y. Thus, Syndrome decoding can be seen as nearest neighbor decoding, which in turn is equivalent to maximum-likelihood decoding. But this approach is inefficient as it constructs H with exponential entries and performs a search over it. However, in the next section, given additional structure, it can be made efficient.*

4.2 Maximum Likelihood Decoding of $\mathrm{RM}(m, m-3)$

Using Proposition 4.5, the parity check matrix of $RM(m, m-3)$ is the generator matrix of $RM(m, 2)$ with dimension $k \times n$, where $k = 1 + m + \binom{m}{2}$ and $n = 2^m$. The number of coset leaders is $2^k = O(n^{(m+1)/2})$ which is super-polynomial in n. Thus, using syndrome decoding (Algorithm 4)

directly, in this case, leads to an impractical algorithm. However, using code symmetry, it is possible to reduce the size of the syndrome table to $O(m)$.

Algorithm 5 Syndrome Decoding for BSC [19]

(a) Compute the syndrome polynomial $[y(x)]_{m-2}^{m}$ using Lemma 4.10.

(b) Perform an affine transform $\pi(A, c)$ according to Lemma 4.11 to obtain $[y(Ax + c)]_{m-2}^{m}$.

(c) Obtain a coset leader e as mentioned in [28, Theorem 2(b)].

(d) Perform inverse of affine transform $\pi(A, c)$ on e.

(e) Output $x = y + e$ as the maximum likelihood estimate of the transmitted codeword.

Suppose $y \in \mathbb{F}^n$ is the received vector. The polynomial corresponding to received vector y is written as:

$$y(x) = y_0 x_{[m]} + \sum_{i=1}^{m} y_i x_{\sim i} + \sum_{i=1}^{m} \sum_{j=i+1}^{m} y_{ij} x_{\sim i,j} + \text{lower-order terms}, \tag{4.2}$$

where $x_S = \prod_{i \in S} x_i$ and $x_{\sim S} = x_{[m] \setminus S}$. The following result relates $y(x)$ to its syndrome.

Lemma 4.10. *Suppose the received vector is represented as the polynomial $y(x)$ shown in* (4.2) *and let $s = Hy^T = (s_0, \ldots, s_m, s_{11}, \ldots, s_{ij}, \ldots, s_{(m-1)m})$, $1 \leq i < j \leq m$, be the syndrome. Then, we have*

$$y_0 = s_0 = v(1)y^T, \quad y_i = s_i = v(1 + x_i)y^T, \quad y_{ij} = s_{ij} = v((1 + x_i)(1 + x_j))y^T.$$

We define the syndrome polynomial $[y(x)]_{m-2}^{m}$ as $y(x)$ ignoring the lower-order terms.

An affine transformation is applied on $y(x)$ such that it gets mapped to one of the entries in the reduced syndrome table.

Lemma 4.11. *Let $e = m \bmod 2$ and $\bar{e} = 1 - e$. For any received vector $y \in \mathbb{F}_2^n$, there exists an affine transformation $\pi(A, c)$ that can be found by performing a maximum of $O(m^3)$ operations such that any non-zero $[y(Ax + c)]_{m-2}^{m}$ is one of the entries in Table 4.1.*

Theorem 4.12. *Syndrome decoding of $RM(m, m-3)$ (Algorithm 5) with blocklength $n = 2^m$ can be performed with $O(m^3)$ operations per codeword using a table with $1.5\,m$ syndrome entries.*

1	$x_{[m-2]}$ $x_{[m-2]} + x_{\sim m-3,m-2}$ $\vdots$ $x_{[m-2]} + x_{\sim m-3,m-2} + \cdots + x_{\sim e+1,e+2}$
2	$x_{[m-1]}$ $x_{[m-1]} + x_{[m-3]}x_m$ $x_{[m-1]} + x_{[m-3]}x_m + x_{\sim m-4,m-3}$ $\vdots$ $x_{[m-1]} + x_{[m-3]}x_m + x_{\sim m-4,m-3} + \cdots + x_{\sim \bar{e}+1,\bar{e}+2}$
3	$x_{[m]}$ $x_{[m]} + x_{[m-2]}$ $x_{[m]} + x_{[m-2]} + x_{\sim m-3,m-2}$ $\vdots$ $x_{[m]} + x_{[m-2]} + x_{\sim m-3,m-2} + \cdots + x_{\sim e+1,e+2}$

Table 4.1: Reduced Syndrome Table

4.3 Recursive Puncturing–Aggregation Decoding (RXA)

Recursion-based decoding algorithms using the symmetry of RM codes were recently introduced - Recursive Puncturing–Aggregation (RXA) Decoding [18], Recursive Projection–Aggregation (RPA) Decoding [17], to name a few. Next, we provide the details of the RXA algorithm, which will be used in building our proposed algorithm.

The RXA decoding algorithm is presented in Algorithm 6. As the name suggests, this decoder punctures an $RM(m, r)$ received vector into two halves of $RM(m-1, r)$ recursively. It computes all possible permutations of the received vector, which result in new codewords plus some error. The base of recursion is $RM(r+2, r)$, which is solved using the extended Hamming FHT decoder [29] to obtain an estimated codeword. At each level, there are $2(n-1)$ estimated codewords corresponding to the permutations. Using a majority voting aggregation, an estimated codeword is computed for each level. As we have permuted the received vector y into y_{π_i} for each permutation π_i, each position z in y corresponds to a different position in the permuted vector. So, during aggregation, we need a mapping between the positions in the received and the permuted vectors. This can be seen in line-4, where FindIndex(z, i) gives this mapping corresponding to i^{th} permutation. For each position $z \in \mathbb{F}_2^m$ of the codeword, we iterate through all $(n-1)$ permutations. So, we can observe that the maximum value of $changevote(z)$ for each $z \in \mathbb{F}_2^m$ is $(n-1)$. The algorithm iterates for N_{max} rounds to ensure the convergence to a fixed point.

Algorithm 6 The RXA decoding function for BSC

Input: The corrupted codeword $y = (y(z), z \in \mathbb{F}_2^m)$; Reed-Muller parameter m and r.

Output: The decoded codeword $\hat{c}$

for $j = 1, 2, \ldots, N_{\text{max}}$ **do** ▷ $N_{max} = \lfloor m/2 \rfloor$
  if $m = r + 2$ **then**
    $\hat{c} = \text{decode}(y, m, r)$ ▷ Decode using extended Hamming FHT Decoder [29]
  else
    for $i = 1, 2, \ldots, 2^m - 1$ **do**
      $y_{\pi_i} \longleftarrow \text{Permutation}(y, i)$ ▷ π_i is a permutation from $\text{Aut}(RM(m, r))$
      $y_{/1,i} \longleftarrow y_{\pi_i}[1 : 2^{m-1}]$ ▷ First half of y_{π_i}
      $y_{/2,i} \longleftarrow y_{\pi_i}[2^{m-1} + 1 : 2^m]$ ▷ Second half of y_{π_i}
      $\hat{y}_{/j,i} \longleftarrow \text{RXA}(y_{/j,i}, m - 1, r)$ for $j = 1, 2$
    end for
    $\hat{y} \longleftarrow \text{Aggregation}(y, \hat{y}_{/1,1}, \hat{y}_{/2,1}, \ldots, \hat{y}_{/1,n-1}, \hat{y}_{/2,n-1})$
  end if
end for
$\hat{c} \longleftarrow \hat{y}$

Algorithm 7 The Aggregation function in RXA

Input: $y, \hat{y}_{/1,1}, \hat{y}_{/2,1}, \ldots, \hat{y}_{/1,n-1}, \hat{y}_{/2,n-1}$

Output: The estimated codeword $\hat{y}$

Initialize $(\text{changevote}(z), z \in \{0, 1\}^m)$ as an all-zero vector indexed by $z \in \{0, 1\}^m$
for $z = 1, 2, \ldots, 2^m$ **do** ▷ Iterate through all the positions of codeword
  for $i = 1, 2, \ldots, 2^m - 1$ **do** ▷ Iterate through estimates of all permutations
    $z' \longleftarrow \text{FindIndex}(z, i)$
    $\text{changevote}(z) += \mathbf{1}[y(z) \neq \hat{y}_{/j,i}(z'')], j = 1 + \lfloor \frac{z'}{(n/2)} \rfloor, z'' = z' \% (n/2)$
  end for
end for
$y(z) = y(z) + \mathbf{1}[\text{changevote}(z) > (n - 1)/2]$ for each $z = 1, 2, \ldots, 2^m$
$\hat{y} \longleftarrow y$

4.4 RXA-Syndrome decoding of $\mathrm{RM}(m, m-4)$

Algorithm 8 The RXA-SYNDROME decoding function for BSC

Input: The corrupted codeword $y = (y(z), z \in \mathbb{F}_2^m)$; Reed-Muller parameter m, r.

Output: The decoded codeword $\hat{c}$

1: **for** $j = 1, 2, \ldots, N_{\text{max}}$ **do** ▷ $N_{max} = \lfloor m/2 \rfloor$
2: **if** $r = m - 3$ **then**
3: $\hat{c} =$ Syndrome-decode$(y, m, m-3)$ ▷ Decode using Syndrome Decoding
4: **else**
5: **for** $i = 1, 2, \ldots, 2^m - 1$ **do**
6: $y_{\pi_i} \longleftarrow \mathrm{Permutation}(y, i)$ ▷ π_i is a permutation from $\mathrm{Aut}(RM(m, r))$
7: $y_{/1,i} \longleftarrow y_{\pi_i}[1 : 2^{m-1}]$ ▷ First half of y_{π_i}
8: $y_{/2,i} \longleftarrow y_{\pi_i}[2^{m-1} + 1 : 2^m]$ ▷ Second half of y_{π_i}
9: $\hat{y}_{/j,i} \longleftarrow \mathrm{RXA-SYNDROME}(y_{/j,i}, m-1, r)$ for $j = 1, 2$
10: **end for**
11: $\hat{y} \longleftarrow \mathrm{Aggregation}(y, \hat{y}_{/1,1}, \hat{y}_{/2,1} ..., \hat{y}_{/1,n-1}, \hat{y}_{/2,n-1})$
12: **end if**
13: **end for**
14: $\hat{c} \longleftarrow \hat{y}$

The pseudo-code is presented in Algorithm 8. There are $2^m - 1$ permutations in the automorphism group of an $RM(m, r)$ code. After choosing a permutation π_i, the received vector is permuted accordingly to y_{π_i}. This permuted vector is divided into two halves, where each half corresponds to a corrupted codeword of $RM(m-1, m-4)$ code. As shown in Fig. 4.1, each of these halves is decoded using the maximum likelihood decoder (Algorithm 5). The estimates for these halves are denoted by $\hat{y}_{/1,1}, \hat{y}_{/2,1}, \ldots, \hat{y}_{/1,n-1}, \hat{y}_{/2,n-1}$. All these estimates corresponding to the $2^m - 1$ permutations are used in the Aggregation step (Algorithm 7). In order to get a new estimate $\hat{y}$ of the original codeword, a majority voting scheme is used to aggregate the decoding results.

The computational complexity of syndrome decoding for $RM(m, m-3)$ codes is $O(m^3)$. The above procedure is repeated for $(2^m - 1)$ permutations which belong to the automorphism group of the $RM(m, m-4)$ code. For each permutation, the computational complexity of decoding is $2 \times O((m-1)^3)$. So, the overall computational complexity of the algorithm is $(2^m - 1) \times 2 \times (m-1)^3$, which is $O(2^m m^3)$.

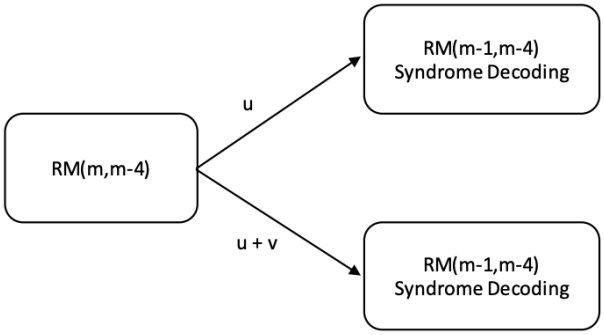

Figure 4.1: RXA-Syndrome Decoding of $RM(m, m-4)$ code.

In low and high code rate regimes, the performance of the RPA algorithm [17] is close to that of the maximal likelihood decoder. Due to this reason, we compare the proposed algorithm against RPA. The complexity of RPA [17] is $O(n^r logn)$ which is equivalent to $O(2^{m^2} \times m)$. RXA-Syndrome is faster when compared to RPA. The simulation results are shown in Fig. 4.2, where we compute the frame error rate (FER) in decoding $RM(6, 2)$ and $RM(7, 3)$ codes. We use BSC with transition probability p for different values. It can be seen that RPA has better performance than RXA-Syndrome. The performance deficit of RXA-Syndrome can be explained as follows. In [18, Section V. B], it is mentioned that soft-decision decoding is required for base codes of RXA. Similarly, in our simulation results, we noticed a big drop in the performance of RXA for hard-decision when compared to that of soft-decision. Since hard-decision decoding is required to use the syndrome decoding, RXA-Syndrome performs poorly.

Here, we have demonstrated a prototype decoder for $RM(m, m-4)$ codes that use RXA to get to $RM(m-1, m-4)$ codes and apply syndrome decoding to it. To improve the performance, we have to find a replacement for the RXA decoding algorithm, which overcomes the disadvantages of RXA for BSC while maintaining similar complexity. We leave it as a future work.

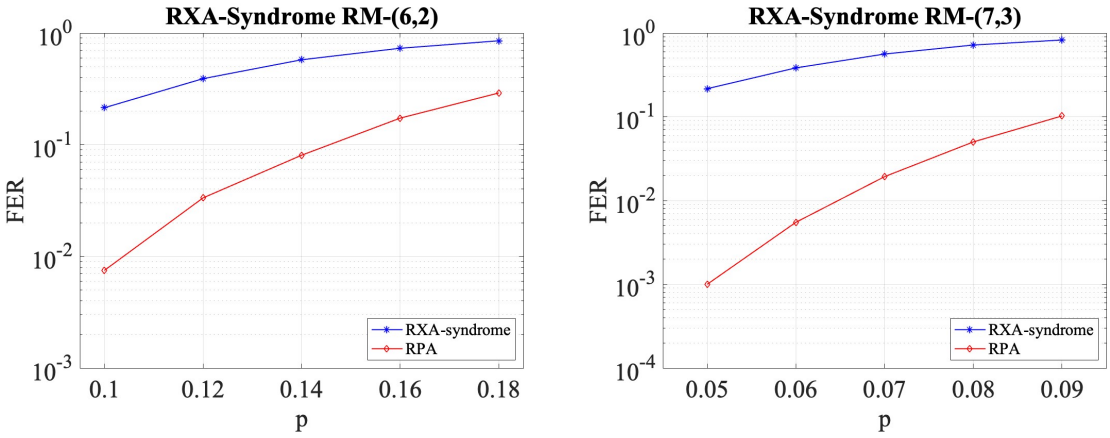

Figure 4.2: Comparing decoding performance of RXA-Syndrome and RPA for $RM(m, m-4)$ codes.

Chapter 5

Structural Properties of Higher-Order MDS codes

5.1 Introduction

The linear codes can recover any pattern of errors, given that the rate R and the fraction of errors ρ satisfy $\rho \leq \frac{1-R}{2}$. This thread of research was initiated by Hamming [30] in 1950, focusing mainly on developing code constructions. Contrary to this worst-case model, Shannon, in his 1948 landmark paper [31], considered the channel transmission problem by modeling the errors as a stochastic process. Shannon was able to deduce the largest possible rate for reliable communication while decoding ρ fraction of errors such that $H_q(\rho) \leq 1-R$. This is roughly twice as many errors recovered in Hamming's model for large q.

This gap can be bridged by extending the notion of unique decoding to list decoding. Next, we define list decodable codes.

A vector $v \in \mathbb{F}^n$ is said to be t-close to $w \in \mathbb{F}^n$ if $d_H(v, w) \leq t$, where $d_H(v, w)$ denotes the hamming distance between v and w. We denote the support of a vector v as $\mathsf{supp}(v)$.

Definition 5.1 (List Decodable code)**.** *For positive integers $t \in [n]$ and L, an $[n, k]$ (linear) code C in $\mathbb{F}^n$ is said to be (t, L)-list decodable if, for any vector $y \in \mathbb{F}^n$, there are at most L codewords which are t-close to y.*

The following result establishes the existence of list decodable codes of moderate list sizes with rates close to $1 - H(\rho)$ where ρ is the fraction of errors.

Proposition 5.2 ([32])**.** *For $\rho \in (0, \frac{1}{2})$ and $L \geq 1$, for all large enough n, there is a $(\rho n, L)$-list decodable binary code of block length n and rate at least $1 - H(\rho) - 1/L$.*

Thus, it is important to study the properties of list decodable codes. In the next section, we present the necessary and sufficient conditions for list decodability.

5.2 Necessary and Sufficient Conditions

Definition 5.3. *Let A be a finite set of vectors from a vector space V over a finite field $\mathbb{F}$. The span of vectors in A is denoted by $\mathsf{sp}(A)$. We define the strict-span of A as*

$$\mathsf{sp}_{\mathsf{str}}(A) \triangleq \left\{ \sum_{v \in A} \alpha_v v : \forall \alpha_v \in \mathbb{F} \setminus \{0\} \right\}.$$

Let M be a matrix with n columns indexed by $[n]$. For a subset $T \subset [n]$, the submatrix with columns indexed by T is denoted by M_T. The span of the columns of M_T is denoted by $\mathsf{sp}(M_T)$.

Theorem 5.4. *If C is (t, L)-list decodable code, then the following conditions should hold.*

- *the rank of any $t+\lfloor t/L \rfloor$ columns of a parity-check matrix H for C is at least $t+\lfloor t/L \rfloor - \lfloor \log_q L \rfloor$.*
- *For any set of $L+1$ distinct $t_{\leq}$-subsets (i.e., subsets of size $\leq t$) of $[n]$ denoted by $T_i : i \in [L+1]$, if there exists some vector $s \in \cap_{i=i}^{L+1} \mathsf{sp}(H_{T_i})$, then $s \notin \mathsf{sp}_{\mathsf{str}}(H_{T_i})$ for at least one $i \in [L+1]$.*

Proof. Suppose the code is list decodable, and the first condition does not hold. That is, there is some $t + \lfloor t/L \rfloor$ columns (indexed by T) such that $\mathsf{rk}(H_T) < t + \lfloor t/L \rfloor - \lfloor \log_q L \rfloor$. Then, we have that $\dim(\mathsf{null}(H_T)) \geq \lfloor \log_q L \rfloor + 1$, which implies $\dim(\mathsf{null}(H_T)) > \log_q L$. Thus, there are at least $L+1$ distinct codewords $c_i : i \in [L+1]$ such that $\mathsf{supp}(c_i) \subseteq T$. Let $T_1 \subset T$ be a $\lfloor t/L \rfloor$-sized subset. We partition the remaining indices of T (except those in T_1) into L subsets of size $\lfloor t/L \rfloor$ or $\lceil t/L \rceil$, denoted by $T_2, \ldots, T_{L+1}$ (note that $\lfloor t/L \rfloor$ could be zero).

Now consider a vector y, which is constructed in the following way:

$$(y)_j = \begin{cases} 0 & \text{if } j \notin T \\ (c_i)_j & \text{if } j \in T_i, \end{cases}$$

where $(y)_j$ and $(c_i)_j$ denote the j^{th} component of y and c_i respectively. That is, y 'agrees' with c_i in the indices in T_i and also everywhere in $[n] \setminus T$.

Thus, $d_H(y, c_i) \leq |T| - |T_i| \leq t + \lfloor t/L \rfloor - \lfloor t/L \rfloor = t$, for each $i \in [L+1]$. This means that the code is not (t, L)-list decodable, in contradiction with our assumption. This completes the proof that the first condition is necessary.

Now, suppose the code is (t, L)-list decodable, and the second condition doesn't hold. That is, there are $L+1$ distinct $t_{\leq}$-subsets, $T_i : i \in [L+1]$, such that there exists a vector $s \in \cap_{i=i}^{L+1} \mathsf{sp}_{\mathsf{str}}(H_{T_i})$.

This implies that there must be distinct vectors $e_i : i \in [L+1]$ with $\mathsf{supp}(e_i) = T_i$ such that $H_{T_i} e_i = s, \forall i \in [L+1]$.

Now consider any codeword $c \in C$. Let $y = c + e_1$. Consider, for $i \in [L+1]$, the vectors $c_i \triangleq y - e_i$. As $Hc_i = Hy - He_i = \mathbf{0}$, we see that $c_i \in C, \forall i$. Also, all the $c_i : i \in [L+1]$ are easily seen to be distinct. Further, observe that $d_H(y, c_i) = \mathrm{wt}(e_i) = |T_i| \leq t$. Thus, we have a contradiction with the (t, L)-list decodability of C, which completes the proof. □

We now prove a sufficient condition for (t, L)-list decodability.

Theorem 5.5. *A code C is (t, L)-list decodable if the following conditions hold for any parity-check matrix H for C.*

- *Every set of t columns of the H matrix are linearly independent, and*
- *For any set of $L+1$ distinct $t_{\leq}$-subsets (i.e., subsets of size $\leq t$) of $[n]$ denoted by $T_i : i \in [L+1]$, if there exists some vector $s \in \cap_{i=i}^{L+1} \mathsf{sp}(H_{T_i})$, then $s \notin \mathsf{sp}_{\mathsf{str}}(H_{T_i})$ for at least one $i \in [L+1]$.*

Proof. Suppose the conditions hold, but the code is not (t, L)-list decodable. Then there exists some vector y such that there are distinct codewords $c_i : i \in [L+1]$ with the condition $d_H(y, c_i) \leq t$. Let $e_i = y - c_i$ (thus, $e_i : i \in [L+1]$ are distinct), and $T_i = |\mathsf{supp}(e_i)|$. Note that $He_i = Hy$, for all i. Thus, we see that $Hy \in \cap_{i=1}^{L+1} \mathsf{sp}_{\mathsf{str}}(H_{T_i})$. Clearly, $|T_i| = \mathsf{supp}(e_i) \leq t$. Further $|T_i \cup T_j| > t$ for any distinct i, j. This must be true, as otherwise, we would have that the columns $T_i \cup T_j$ are linearly dependent (since $H(e_i - e_j) = \mathbf{0}$), which would then violate the first condition in the statement of the theorem. Thus, $T_i \neq T_j$, for all distinct i, j. However, this leads to the situation that the second condition of the theorem is violated, which leads to a contradiction in our assumption that both conditions hold. This completes the proof. □

5.3 Higher-Order MDS Codes

MDS codes achieve the optimal trade-off between the size of the code and its minimum distance for a fixed message and block lengths. The extension of this notion to list decodable codes is of particular interest. A generalization of Singleton bound was proved in [20].

Proposition 5.6 (Generalized Singleton Bound [20])**.** *An $[n, k]_q$ code C that is (t, L)-list decodable satisfies*

$$|C| \leq Lq^{n - \left\lfloor \frac{(L+1)t}{L} \right\rfloor}.$$

In particular, if C is a linear code over $\mathbb{F}_q$ with $q > L$, then $|C| \leq q^{n-\left\lfloor \frac{(L+1)t}{L} \right\rfloor}$.

Note that setting $L = 1$ reduces to the Singleton bound. The generalizations of MDS codes were considered in [21, 22, 33], which we recall below.

In [21], a generalization of MDS codes was considered as the codes that meet this generalized Singleton bound with equality. This generalization uses a stronger notion of list decodability called average-radius list decodable code.

Definition 5.7 ([34])**.** *Let $0 \leq \rho \leq 1$ and $L \geq 1$. A code C is $(\rho n, L)$-average-radius list decodable if for all sets S of codewords with cardinality at least L, and for all $x \in \mathbb{F}^n$, we have*

$$\frac{1}{|S|} \sum_{c \in S} d_H(c, x) \geq \rho.$$

In [21], they considered the average-radius list decodable codes that meet generalized Singleton bound with equality to be a suitable extension of MDS codes.

Definition 5.8 (List Decodable-MDS code)**.** *An $[n, k]_q$ code C is defined as list decodable-MDS(L) (or simply LD-MDS(L)), if C is $(\rho = \frac{L}{L+1}(1 - k/n), L)$ average-radius list decodable. In other words, C meets the Generalized Singleton Bound with equality.*

Note that an MDS code is an LD-MDS(1) code. In [22, 33], another generalization of MDS codes was considered, which uses the notion of a generic collection.

Definition 5.9 (Generic Collection)**.** *Let ℓ be a positive integer and let n, k be integers such that $n \geq k \geq 0$. Consider $A_1, \ldots, A_\ell \subseteq [n]$ with $|A_i| \leq k$. These sets $(A_1, \ldots, A_\ell)$ are called an (n, k, ℓ)-generic collection if for all partitions $P_1 \cup \cdots \cup P_s = [\ell]$ we have*

$$\sum_{i=1}^{s} \left| \bigcap_{j \in P_i} A_j \right| \leq (s-1)k. \tag{5.1}$$

In the content that follows, for a matrix V, we denote by V_A the span of the submatrix of V consisting of the columns in A.

Definition 5.10. *(Higher order MDS codes [22]) An $[n, k]_q$ code C with generator V is called an (n, k)-MDS(ℓ) code if for all (n, k, ℓ)-generic collections $A_1, \ldots, A_\ell$ with $|A_1| + \cdots + |A_\ell| = (\ell - 1)k$, we have*

$$\bigcap_{i=1}^{\ell} V_{A_i} = \mathbf{0}. \tag{5.2}$$

Remark 5.11. *Consider $[n,k]_q$ MDS code C with generator matrix V. Then, any k columns of V are linearly independent. This can be equivalently expressed as: For any disjoint subsets A_1 and A_2 of $[n]$, where $|A_1|+|A_2|=k$, the intersection of the subspaces V_{A_1} and V_{A_2} is the zero subspace, i.e., $V_{A_1}\cap V_{A_2}=\mathbf{0}$. Thus, an MDS code is an MDS(2) code.*

The two generalizations of MDS codes are related by duality, as shown below.

Proposition 5.12 ([33])**.** *An MDS(ℓ) code is the dual of a code that is LD-MDS(L) for all $L\leq \ell-1$.*

Using this duality, the structural results on higher-order MDS codes can be translated to those of LD-MDS codes.

Understanding the higher-order MDS codes is crucial to benefit from them. The existing structural result on higher-order MDS code is given below.

Lemma 5.13 ([22])**.** *An (n,k)-MDS(ℓ) code C is also an (n,k)-MDS($\ell-1$) code for all $\ell\geq 3$.*

5.4 Structural Results

In this section, we present more structural results which help us better understand higher-order MDS codes. These results will be useful in obtaining constructions and further developing efficient list-decoding algorithms, which we leave as future work.

Theorem 5.14. *An (n,k)-MDS(k) code is also an (n,k)-MDS(ℓ) code for all $\ell\geq k$.*

Proof. From Definition 5.10, we know that V is (n,k)-MDS(ℓ) if and only if for all (n,k,ℓ)-generic collection of subsets $(A_1,\ldots,A_\ell): A_i\subseteq[n],\forall i$ with

$$|A_1|+\cdots+|A_\ell|=(\ell-1)k, \tag{5.3}$$

we have that

$$V_{A_1}\cap\cdots\cap V_{A_\ell}=\mathbf{0}. \tag{5.4}$$

We prove that V is (n,k)-MDS(ℓ) for all $\ell\geq k$ by showing that (5.4) is satisfied for each possible (n,k,ℓ)-generic collection satisfying (5.3).

Observe that when any of the A_i's are empty, then (5.4) holds trivially. Now, we look at all non-zero possibilities for the cardinality of all A_is in an (n,k,ℓ)-generic collection and check if (5.3) holds:

- Suppose all A_i's are of size k, then

$$|A_1|+\cdots+|A_\ell|=\ell k.$$

But from Definition 5.10, we want $|A_1|+\cdots+|A_\ell|=(\ell-1)k$. So, all A_i's cannot be of size k.

- Suppose, exactly $(\ell-1)$ of the A_i's are of size k. WLOG, say $A_1, \ldots, A_{\ell-1}$ are of size k. Then

$$|A_1| + \cdots + |A_{\ell-1}| = (\ell-1)k. \tag{5.5}$$

From (5.3) and (5.5), we get $|A_\ell| = 0$, and this scenario is already handled. So, we cannot have $(\ell-1)$ of the A_i's of size k.

- For $2 \leq x \leq k$, suppose $(\ell-x)$ of the A_i's are of size k. WLOG, say $A_1, \ldots, A_{\ell-x}$ are of size k. Then

$$|A_1| + \cdots + |A_{\ell-x}| = (\ell-x)k. \tag{5.6}$$

From (5.3) and (5.5), we get

$$|A_{\ell-x+1}| + \cdots + |A_\ell| = (x-1)k. \tag{5.7}$$

Let $\{Q_1, \ldots, Q_{s'}\}$ be any arbitrary partition of $[\ell] \setminus [\ell-x]$. Consider the following partition of $[\ell]$ given as $\{P_i = \{i\} : i \in [\ell-x]\} \cup \{Q_1, \ldots, Q_{s'}\}$. As $\{A_1, \ldots, A_\ell\}$ is (n,k,ℓ)-generic collection of subsets of $[n]$, we have that

$$\begin{aligned}
\sum_{i=1}^{\ell-x} \left| \bigcap_{j \in P_i} A_j \right| + \sum_{i=1}^{s'} \left| \bigcap_{j \in Q_i} A_j \right| &\leq (\ell-x+s'-1)k \\
(\ell-x)k + \sum_{i=1}^{s'} \left| \bigcap_{j \in Q_i} A_j \right| &\leq (\ell-x+s'-1)k \\
\sum_{i=1}^{s'} \left| \bigcap_{j \in Q_i} A_j \right| &\leq (s'-1)k.
\end{aligned} \tag{5.8}$$

Using (5.8) we can say that $A_{\ell-x+1}, \ldots, A_\ell$ are a (n,k,x)-generic collection of subsets of $[n]$. As V is MDS(k) it is also MDS(x) as $x \leq k$, which implies $V_{A_{\ell-x+1}} \cap \cdots \cap V_{A_\ell} = \mathbf{0}$. Therefore $V_{A_1} \cap \cdots \cap V_{A_\ell} = \mathbf{0}$.

- For $k < x \leq \ell$, if $(\ell-x)$ of the A_i's are of size k, WLOG say $A_1, \ldots, A_{\ell-x}$ are of size k. Then

$$|A_1| + \cdots + |A_{\ell-x}| = (\ell-x)k. \tag{5.9}$$

Using (5.3) and (5.9), we get

$$|A_{\ell-x+1}| + \cdots + |A_\ell| = (x-1)k = xk - k.$$

But,

$$|A_{\ell-x+1}| + \cdots + |A_\ell| \leq x(k-1) = xk - x$$

(as each A_i, $i \in [\ell] \setminus [\ell - x]$ is of size at most $k-1$). As $xk - x < xk - k$, we get a contradiction. Hence, we cannot have $(\ell - x)$ A_is of size k when $x > k$.

From above, we have that for all $A_1, \ldots, A_\ell \subseteq [n]$ with $|A_i| \leq k$ and $|A_1| + \cdots + |A_\ell| = (\ell - 1)k$, we have $V_{A_1} \cap \cdots \cap V_{A_\ell} = \mathbf{0}$ as long as V is (n, k)-MDS(k). Hence, V is (n, k)-MDS(ℓ) code for all $\ell \geq k$. □

Using Theorem 5.14, one can focus on the construction of (n, k)-MDS(ℓ) for $\ell \leq k$ and consequently, obtain the codes for $\ell \geq k$. Combining the results in Lemma 5.13 and Theorem 5.14, we have the following corollary.

Corollary 5.15. *For $\ell \geq k$, an $[n, k]_q$ code is an (n, k)-MDS($\leq \ell$) if and only if it is (n, k)-MDS(k).*

Definition 5.16. *An (n, k, ℓ)-generic collection $(A_1, \ldots, A_\ell)$ is said to be an $(n, k, \ell)_{k-1}$-generic collection if $|A_i| = k - 1$, for all $i \in [\ell]$.*

Theorem 5.17. *Let V be an (n, k)-MDS($k-1$) code. Then V is MDS(k) if and only if for each $(n, k, k)_{k-1}$-generic collection $(A_1, \ldots, A_k)$, we have $\cap_{i=1}^{k} V_{A_i} = \mathbf{0}$.*

Proof. From Definition 5.10, we know that V is (n, k)-MDS(k) if and only if for all (n, k, k)-generic collection of subsets $A_1, \ldots, A_k$ with

$$|A_1| + \cdots + |A_k| = (k-1)k, \tag{5.10}$$

we have that

$$V_{A_1} \cap \cdots \cap V_{A_k} = \mathbf{0}. \tag{5.11}$$

Now, we want to show that, given V is (n, k)-MDS($k-1$), it suffices to check $V_{A_1} \cap \cdots \cap V_{A_k} = \mathbf{0}$ when A_i's are a $(n, k, k)_{k-1}$-generic collection for V to be (n, k)-MDS(k).

Observe that when any of the $A_i, i \in [k]$ are empty, then (5.11) holds trivially. Now, if we look at all non-zero possibilities for the cardinality of A_is in a generic collection and check if (5.11) holds:

- Suppose all A_i's are of size k, then

$$|A_1| + \cdots + |A_\ell| = \ell k.$$

But from Definition 5.10, we want $|A_1| + \cdots + |A_\ell| = (\ell - 1)k$. So, all A_i's cannot be of size k.

- Suppose, exactly $(k-1)$ of the A_i's are of size k, WLOG say $A_1, ...A_{k-1}$ are of size k. Then

$$|A_1| + \cdots + |A_{k-1}| = (k-1)k. \tag{5.12}$$

From (5.10) and (5.11), we get $|A_k| = 0$, a contradiction to our assumption that none of the $A_i, i \in [k]$ have size 0. Thus, we cannot have $(k-1)$ of the A_i's of size k.

- For $2 \leq x \leq k-1$, suppose $(k-x)$ of the A_is are of size k, WLOG say $A_1, \ldots, A_{k-x}$ are of size k. Then

$$|A_1| + \cdots + |A_{k-x}| = (k-x)k \tag{5.13}$$

From (5.10) and (5.13), we get

$$|A_{k-x+1}| + \cdots + |A_k| = (x-1)k. \tag{5.14}$$

Let $\{Q_1, \ldots, Q_{s'}\}$ be any arbitrary partition of $[k] \setminus [k-x]$. Consider the following partition of $[k]$ given as $\{P_i = \{i\} : i \in [k-x]\} \cup \{Q_1, \ldots, Q_{s'}\}$. As $(A_1, \ldots, A_k)$ is an (n,k,k)-generic collection of subsets of $[n]$, we have that

$$\begin{aligned} \sum_{i=1}^{k-x} \left| \bigcap_{j \in P_i} A_j \right| + \sum_{i=1}^{s'} \left| \bigcap_{j \in Q_i} A_j \right| &\leq (k-x+s'-1)k \\ (k-x)k + \sum_{i=1}^{s'} \left| \bigcap_{j \in Q_i} A_j \right| &\leq (k-x+s'-1)k \\ \sum_{i=1}^{s'} \left| \bigcap_{j \in Q_i} A_j \right| &\leq (s'-1)k. \end{aligned} \tag{5.15}$$

Using (5.15) we can say that $(A_{k-x+1}, \ldots, A_k)$ is an (n,k,x)-generic collection of subsets of $[n]$. As V is MDS$(k-1)$ it is also MDS(x), which implies $V_{A_{k-x+1}} \cap \cdots \cap V_{A_k} = \mathbf{0}$. Therefore $V_{A_1} \cap \cdots \cap V_{A_k} = \mathbf{0}$ is satisfied automatically by the given condition.

- Suppose, there are no $A_i, i \in [k]$ of size k. Then, each of the $A_i, i \in [k]$ are of size $\leq (k-1)$. In order to satisfy (5.10), the only possibility is each $A_i, i \in [k]$ must have cardinality equal to $k-1$. Note that this means $(A_i : i \in [k])$ is an $(n,k,k)_{k-1}$-generic collection. Therefore, for V to be (n,k)-MDS(k) it suffices to check $V_{A_1} \cap \cdots \cap V_{A_k} = \mathbf{0}$ for each $(n,k,k)_{k-1}$-generic collection $(A_1, \ldots, A_k)$.

□

To show that a code is (n, k)-MDS(k), instead of checking the condition of trivial intersection for all (n, k, k)-generic collections, we only need to check it for all $(n, k, k)_{k-1}$-generic collections. This simplifies the conditions in comparison to the one in Definition 5.10, which in turn helps in the construction of new MDS(k) codes from existing higher order MDS codes.

Chapter 6

Conclusion

In this thesis, we have considered three problems related to coding theory. In the first problem, we have considered a distributed multi-user secret sharing (DMUSS) system, where the dealer conveys a specific subset of secrets to each user via the storage nodes. Our main result is the design of optimal weakly secure DSSP and compares its performance using well-known constructions for t-disjunct matrices. We characterized the capacity region of the DMUSS considered. An interesting direction for future work is to design a perfectly secure DSSP at the cost of increased storage overhead (due to the trade-off between SO and security level [1]).

In the second problem, we proposed an RXA-Syndrome decoding algorithm for $RM(m, m-4)$ code that exploits the large symmetry in the automorphism group of RM codes and the optimality of the maximum likelihood decoder. As a future work, one can extend the syndrome decoding algorithm from $RM(m, m-3)$ to $RM(m, m-4)$. Another direction could be to find replacements for the RXA decoding algorithm (to get $RM(m, m-3)$ codes from $RM(m, m-4)$ codes so that syndrome decoding of $RM(m, m-3)$ can be performed), which can improve the performance compared to our proposed algorithm.

In the last problem, we considered the list decodable codes and the higher order MDS codes, a special class of list decodable codes. We have derived the necessary and sufficient conditions for list decodability. We have presented several new structural results of higher order MDS codes. We showed that an (n, k)-MDS(k) code is also (n, k)-MDS(ℓ) for all $\ell > k$. We have simplified the conditions under which an (n, k)-MDS($k-1$) code is also an (n, k)-MDS(k) code. As a future work, these structural results on higher order MDS codes can be useful in obtaining explicit constructions for constant list sizes. Further, they can be used in developing efficient list-decoding algorithms.

Publications

1. Rasagna Chigullapally, Harshitanjani Athi, V. Lalitha, Nikhil Karamchandani. "On Distributed Multi-User Secret Sharing with Multiple Secrets per User" accepted to *30th National Conference on Communications (NCC), 2024.*

2. Harshitanjani Athi, Rasagna Chigullapally, Prasad Krishnan and V. Lalitha, "On the Structure of Higher Order MDS Codes," 2023 IEEE International Symposium on Information Theory (ISIT), Taipei, Taiwan, 2023, pp. 1009-1014, doi: 10.1109/ISIT54713.2023.10206712.

Bibliography

[1] M. Soleymani and H. Mahdavifar, "Distributed multi-user secret sharing," *IEEE Transactions on Information Theory*, vol. PP, 01 2018.

[2] G. Reeves and H. D. Pfister, "Reed–muller codes on bms channels achieve vanishing bit-error probability for all rates below capacity," *IEEE Transactions on Information Theory*, 2023.

[3] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, p. 612–613, 1979.

[4] G. R. Blakley, "Safeguarding cryptographic keys," *1979 International Workshop on Managing Requirements Knowledge (MARK)*, pp. 313–318, 1899.

[5] R. Cramer, I. Damgård, and U. Maurer, "General secure multi-party computation from any linear secret-sharing scheme," *Advances in Cryptology—EUROCRYPT 2000*, pp. 316–334, 2000.

[6] M. O. Rabin, "Randomized byzantine generals," *24th annual symposium on foundations of computer science (sfcs 1983)*, pp. 403–409, 1983.

[7] S. Takahashi and K. Iwamura, "Secret sharing scheme suitable for cloud computing," *2013 IEEE 27th International Conference on Advanced Information Networking and Applications (AINA)*, pp. 530–537, 2013.

[8] Y. Desmedt, "Some recent research aspects of threshold cryptography," *Information Security: First International Workshop, ISW'97 Tatsunokuchi, Ishikawa, Japan September 17–19, 1997 Proceedings 1*, pp. 158–173, 1998.

[9] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," *Proceedings of the 13th ACM conference on Computer and communications security*, pp. 89–98, 2006.

[10] Y. Liu and Q. Zhao, "E-voting scheme using secret sharing and k-anonymity," *World Wide Web*, vol. 22, pp. 1657–1667, 2019.

[11] R. K. Raman and L. R. Varshney, "Distributed storage meets secret sharing on the blockchain," *2018 information theory and applications workshop (ITA)*, pp. 1–6, 2018.

[12] A. Baccarini, M. Blanton, and C. Yuan, "Multi-party replicated secret sharing over a ring with applications to privacy-preserving machine learning," *Cryptology ePrint Archive*, 2020.

[13] A. Khalesi, M. Mirmohseni, and M. A. Maddah-Ali, "The capacity region of distributed multi-user secret sharing," *IEEE Journal on Selected Areas in Information Theory*, vol. 2, no. 3, pp. 1057–1071, 2021.

[14] I. S. Reed, "A class of multiple-error-correcting codes and the decoding scheme," *IEEE Transactions on Information Theory*, vol. 4, no. 4, pp. 38–49, 1954.

[15] R. Green, "A serial orthogonal decoder," *JPL Space Programs Summary*, vol. 37, pp. 247–253, 1966.

[16] I. Dumer, "Soft-decision decoding of reed-muller codes: a simplified algorithm," *IEEE transactions on information theory*, vol. 52, no. 3, pp. 954–963, 2006.

[17] M. Ye and E. Abbe, "Recursive projection-aggregation decoding of reed-muller codes," *IEEE Transactions on Information Theory*, vol. 66, no. 8, pp. 4948–4965, 2020.

[18] M. Lian, C. Häger, and H. D. Pfister, "Decoding reed–muller codes using redundant code constraints," in *2020 IEEE International Symposium on Information Theory (ISIT)*, pp. 42–47, IEEE, 2020.

[19] A. Thangaraj and H. D. Pfister, "Efficient maximum-likelihood decoding of reed–muller rm (m-3, m) codes," in *2020 IEEE International Symposium on Information Theory (ISIT)*, pp. 263–268, IEEE, 2020.

[20] C. Shangguan and I. Tamo, "Combinatorial list-decoding of reed-solomon codes beyond the johnson radius," in *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pp. 538–551, 2020.

[21] R. M. Roth, "Higher-order MDS codes," *IEEE Transactions on Information Theory*, vol. 68, no. 12, pp. 7798–7816, 2022.

[22] J. Brakensiek, S. Gopi, and V. Makam, "Lower bounds for maximally recoverable tensor codes and higher order mds codes," *IEEE Transactions on Information Theory*, vol. 68, pp. 1–1, 11 2022.

[23] W. Kautz and R. Singleton, "Nonrandom binary superimposed codes," *IEEE Trans. Inf. Theor.*, vol. 10, p. 363–377, sep 2006.

[24] E. Porat and A. Rothschild, "Explicit nonadaptive combinatorial group testing schemes," *IEEE Trans. Inf. Theor.*, vol. 57, p. 7982–7989, dec 2011.

[25] H. A. Inan, P. Kairouz, and A. Özgür, "Sparse combinatorial group testing," *IEEE Transactions on Information Theory*, vol. 66, no. 5, pp. 2729–2742, 2020.

[26] G. G. T. Balint, "Construction in non-adaptive group testing steiner systems and latin squares," *Ph.D. thesis, Illinois Institute of Technology*, 2014.

[27] T. P. Kirkman, "On a problem in combinations," *The Cambridge and Dublin Mathematical Journal*, vol. II, pp. 191–204, 1847.

[28] G. Seroussi and A. Lempel, "Maximum likelihood decoding of certain reed-muller codes (corresp.)," *IEEE Transactions on Information Theory*, vol. 29, no. 3, pp. 448–450, 1983.

[29] A. Ashikhmin, G. Kramer, and S. ten Brink, "Extrinsic information transfer functions: model and erasure channel properties," *IEEE Transactions on Information Theory*, vol. 50, no. 11, pp. 2657–2673, 2004.

[30] R. W. Hamming, "Error detecting and error correcting codes," *The Bell system technical journal*, vol. 29, no. 2, pp. 147–160, 1950.

[31] C. E. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[32] V. Guruswami, "Bridging shannon and hamming: List error-correction with optimal rate," in *Proceedings of the International Congress of Mathematicians 2010 (ICM 2010) (In 4 Volumes) Vol. I: Plenary Lectures and Ceremonies Vols. II–IV: Invited Lectures*, pp. 2648–2675, World Scientific, 2010.

[33] J. Brakensiek, S. Gopi, and V. Makam, "Generic reed-solomon codes achieve list-decoding capacity," *ArXiv*, vol. abs/2206.05256, 2022.

[34] A. Rudra and M. Wootters, "Average-radius list-recoverability of random linear codes," in *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 644–662, SIAM, 2018.