Aalto University
School of Science
Master's Programme in Mathematics and Operations Research

Olga Kuznetsova

Private Information Retrieval:

Combinatorics of the Star-Product Scheme

Master's Thesis
Espoo, June 2, 2019

Supervisor:	Professor Camilla Hollanti
Advisor:	Dr. Ragnar Freij-Hollanti

Aalto University
School of Science
Master's Programme in Mathematics and Operations Research

ABSTRACT OF MASTER'S THESIS

Author:	Olga Kuznetsova		
Title:			
Private Information Retrieval: Combinatorics of the Star-Product Scheme			
Date:	June 2, 2019	**Pages:**	72
Major:	Mathematics	**Code:**	SCI3054
Supervisor:	Professor Camilla Hollanti		
Advisor:	Dr. Ragnar Freij-Hollanti		

In coded private information retrieval (PIR), a user wants to download a file from a distributed storage system without revealing the identity of the file. We consider the setting where certain subsets of servers collude to deduce the identity of the requested file. These subsets form an abstract simplicial complex called the collusion pattern. In this thesis, we study the combinatorics of the general star-product scheme for PIR under the assumption that the distributed storage system is encoded using a repetition code.

Keywords: Private information retrieval, Matroids, Simplicial complexes, Linear codes

Language: English

Acknowledgements

I wish to thank Ragnar Freij-Hollanti and Camilla Hollanti for their support during the writing of this thesis.

Espoo, June 2, 2019

Olga Kuznetsova

Contents

Chapter 1

Introduction

It is a late Monday afternoon and Jane Burn, a financial intelligence officer at The Metropolitan Police Service, receives a tip that Big Corp Inc. is involved in money laundering. She immediately dives into the investigation and decides to request the financial statements of Big Corp and its alleged collaborators. The key to the success of her investigation is the ability to move discreetly without alarming the suspects.

How should she go about accessing the files? Clearly, the financial statements of publicly listed companies are public knowledge. And it is also not surprising that a financial intelligence officer would analyze such reports on a regular basis. Therefore, the only sensitive information is the identities of the suspects.

Unfortunately, Jane cannot trust the cloud storage provider where the financial statements are stored. In fact, she knows that the servers located within close proximity collude by sharing the information about the queries that they receive. Their motivation is innocent enough: they want to optimize service delivery and never share the information with third parties, but this is too important to take the risk.

To her advantage, the provider does not have the capacity to collate the queries across all of its servers. So, as long as she moves swiftly and ensures that partially observed queries are not informative, she will be able to maintain the secrecy of her investigation.

This fictional setting illustrates the motivation and key components of private information retrieval (PIR), whose goal is to allow downloading a file from a distributed storage system without revealing the identity of the file to the storage provider. The flavor of PIR that we are interested in requires that it is impossible to recover the sensitive information from partially observed queries. Therefore, privacy achieved thanks to the clever structure of the queries.

As a first step, Jane needs to choose the structure of the queries that would allow her to retrieve the financial statements of a specific company. This structure cannot remain secret from the storage provider and, in fact, privacy is not necessary when Jane performs non-sensitive investigations, *e.g.,* a regular audit. However, the structure should also enable her to switch to sensitive investigations without the storage provider noticing the switch. In other words, she should be able to send secret queries that should appear indistinguishable from the public queries to the storage provider.

Mathematically, the structure of Jane's public queries is given by a linear code Q, which is a finite-dimensional $\mathbb{F}_q$-vector space. Her privacy challenge satisfies three criteria:

1. once a group of servers colludes, then every server in the group knows exactly which queries every other member of the group receives, *i.e.*, collusion is closed under containment;

2. due to external constraints, a group of colluding servers cannot pass information to other groups, *i.e.*, collusion is not transitive;

3. at least one server actively tries to identify secret queries.

These three properties correspond to the concept of the abstract simplicial complex, which we call the collusion pattern $\mathcal{T}$.

Finally, the structural property of the linear code Q that allows Jane to switch from public to private queries is called the lift $Q^{\mathcal{T}}$ and it is rigorously defined in this thesis for the first time. Essentially, the lift provides privacy by attaching the secret query to the parts of the public query that are not visible to the adversaries. This way the adversary cannot tell the difference between a genuine public message and a modified one that contains sensitive information. At the same time, it is easy to recover the secret query for anyone who can observe the entire message.

One may hypothesize that the operation of lifting is purely combinatorial, *i.e.*, if the combinatorial structures of two different codes Q_1 and Q_2, known as their matroids, are the same, then so will be the matroids of $Q_1^{\mathcal{T}}$ and $Q_2^{\mathcal{T}}$. Our main contribution is to show that this hypothesis is false in full generality and to give sufficient conditions that ensure that $Q_1^{\mathcal{T}}$ and $Q_2^{\mathcal{T}}$ are matroid invariant.

The outline of the work is as follows. We set the stage in Chapters 2 through 4, where we discuss relevant facts about coding theory and private information retrieval. Then, in Chapter 5, we focus on the notion of the lift and give a specific algorithm for construction. Finally, our main result and the description of earlier approaches will be presented in Chapters 6 and Appendix A.

Chapter 2

Coding-Theoretic Preliminaries

The codes discussed in this thesis belong to the family of *error-correcting codes*. Originally, such codes were developed to enable reliable transmission of data over unreliable communication channels such as telephone conversations. For example, if a message consists of a string of 0s and 1s and the signal is weak, some 0s may be interpreted as 1s and vice versa. Error correction methods introduce redundancy in the transmitted message to allow for reconstruction of the original message even if some symbols are corrupted.

We are interested in the application of error-correcting codes to data retrieval. The specifics will be discussed in detail in later sections, but we want to briefly mention the motivation already at this stage.

Distributed storage systems may use a subfamily of error-correcting codes known as erasure-correcting codes to ensure that the data can be recovered if some servers fail. Here the assumption is that some symbols may get erased, but the ones that remain do not contain errors. The goal is to be able to recover the original string even when some symbols are lost. This introduces redundancy in the storage system. We shall illustrate this with the following toy example.

Example 2.0.1. *Assume, we have a file $x = [a, b]$ and three servers. Here are two examples of how to store the file on the servers:*

- ***replication:*** *storing both fragments on each of the servers,* i.e.*, the storage system becomes $\{(a, b), (a, b), (a, b)\}$;*
- ***parity check:*** *storing one fragment on each of two servers and their sum on the third server,* i.e.*, the storage system becomes $\{(a), (b), (a + b)\}$.*

Note that, in the case of replication, we need to store twice as much data as in parity-check coding, so it leads to large storage overhead. On the other hand, under replication, it is possible to retrieve the entire file from just one server and it can tolerate failures on two out of three servers.

While error-correcting codes provide an efficient and reliable means for storing data, their redundancy can also be exploited by the users who want to access the storage system. We will illustrate this with another toy example.

Example 2.0.2. *Assume there are three files a, b and c replicated on two non-colluding servers. In other words, the storage system is* $\{(a, b, c), (a, b, c)\}$ *and the servers do not sure the information about the queries they receive.*

A user wants to retrieve a. Assume the user chooses a vector $(\alpha_1, \alpha_2, \alpha_3) \in \mathbb{F}_2^3$ uniformly at random to retrieve the file. Then she sends the query $(\alpha_1, \alpha_2, \alpha_3)$ to the first server and $(\alpha_1 + 1, \alpha_2, \alpha_3)$ to the second server.

The servers compute the responses by taking

$$\langle(\alpha_1, \alpha_2, \alpha_3), (a, b, c)\rangle := \alpha_1 a + \alpha_2 b + \alpha_3 c$$

and $\langle(\alpha_1+1, \alpha_2, \alpha_3), (a, b, c)\rangle$. Thus, the user can recover a privately by taking

$$\langle(\alpha_1, \alpha_2, \alpha_3), (a, b, c)\rangle - \langle(\alpha_1 + 1, \alpha_2, \alpha_3), (a, b, c)\rangle = a.$$

We will now introduce some facts from coding theory that will be used in our work.

2.1 Error-Correcting Codes

In what follows, we summarize the ideas presented in [31], [5], [1]. A standard assumption in coding theory is that information is encoded using a finite *alphabet* Q with $|Q| = q$, containing a distinguished element 0.

We focus on a specific type of codes, called *block codes*, where the coded information can be divided into blocks of n symbols which can be decoded independently. A block code C is a nonempty subset of Q^n.

The elements of a code are called *codewords*. It is often useful to measure the distance between any two codewords, as it describes the failure tolerance.

Definition 1. *If* $\mathbf{x} \in Q^n$*,* $\mathbf{y} \in Q^n$*, then the* Hamming-distance $d(\mathbf{x}, \mathbf{y})$ *(or just distance) of* $\mathbf{x}$ *and* $\mathbf{y}$ *is defined by*

$$d(\mathbf{x}, \mathbf{y}) := \Big|\{i \,|\, 1 \leq i \leq n,\ x_i \neq y_i\}\Big|.$$

The weight $w(\mathbf{x})$ *of* $\mathbf{x}$ *is defined by*

$$w(\mathbf{x}) := d(\mathbf{x}, \mathbf{0}),$$

where $\mathbf{0}$ *is always chosen to be the all-zeros string.*

Observe that the Hamming distance is a metric as it satisfies the triangular inequality, *i.e.*, $d(\mathbf{a}, \mathbf{b}) + d(\mathbf{b}, \mathbf{c}) \geq d(\mathbf{a}, \mathbf{c})$ for all $\mathbf{a}, \mathbf{b}, \mathbf{c} \in Q^n$. We refer the reader to [31] for the discussion of different distance functions.

Definition 2. *For* $\mathbf{x} \in C$*, the* support *is* $\mathrm{supp}(\mathbf{x}) = \{i \in [n] : x_i \neq 0\}$ *and the support of the code* C *is given by* $\mathrm{supp}(C) = \bigcup_{\mathbf{x} \in C} \mathrm{supp}(\mathbf{x})$*.*

It follows from the definitions that $w(\mathbf{x}) = |\,\mathrm{supp}(\mathbf{x})|$.

We can now present a model for information transmission with additive errors. Assume a codeword $\mathbf{x}$ is sent over an unreliable communication channel. Then the received codeword becomes $\mathbf{r} = \mathbf{x} + \mathbf{e}$, where $\mathbf{e} \in Q^n$ is the error introduced during the transmission. The goal of the receiver is to identify the original codeword $\mathbf{x}$. In our work, we are going to assume that this is achieved using *minimal distance decoding*. This means that the receiver tries to identify a codeword $\mathbf{x}' \in C$ such that $d(\mathbf{x}', \mathbf{r})$ is minimal. Note that there can be more than one codeword in C that satisfies this condition.

Our approach makes several implicit assumptions:

- an error in position i is independent of the errors in any other position;
- the symbol in error can be any of the other $q - 1$ elements;
- during communication, all codewords are equally likely.

Earlier we defined distance and weight for individual codewords. We can now consider similar notions for entire codes.

Definition 3. *The* minimum distance *of a code* C *is*

$$\min\{d(\mathbf{x}, \mathbf{y}) \,|\, \mathbf{x} \in C, \mathbf{y} \in C, \mathbf{x} \neq \mathbf{y}\}.$$

It is usually denoted by $d_{\min}$ or just d when the reference to minimum distance is clear from the context.

Definition 4. *The* minimum weight *of C is*

$$\min\{w(\mathbf{x}) \,|\, \mathbf{x} \in C, \mathbf{x} \neq \mathbf{0}\}.$$

Similarly, it is denoted by $w_{\min}$ or just w when no confusion can arise.

An elementary but useful observation is that if there are at most e errors in the received codeword $y \in Q^n$ and the minimum distance is $d_{\min} = 2e+1$, then the receiver will always have a unique matching codeword $x \in C$. This is because if $d(x,y) = e$ and for all codewords $x' \in C$ different from x, $d(x,x') \geq 2e+1$, then $d(x',y) \geq e+1$. A code where every $y \in Q^n$ has distance $\leq e$ to exactly one codeword in C is called *perfect*.

Finally, it is often useful to evaluate the efficiency of a given code using the information rate.

Definition 5. *If $|Q| = q$ and $C \subset Q^n$, then*

$$R := n^{-1} \log_q |C|$$

is called (information) rate *of C.*

2.2 Linear Codes

We now turn our attention to a special type of block codes that have nice algebraic properties. We let $Q = \mathbb{F}_q$, where $\mathbb{F}_q$ is a finite field and, hence, $q = p^n$, where p is a prime.

Definition 6. *A q-ary* linear code *C is a linear subspace of $\mathbb{F}_q^n$. If C has dimension k and minimum distance d, then C is called an $[n,k,d]$ code.*

The rate of an $[n,k,d]$ code C is $\frac{k}{n}$. One advantage of linear codes is that it is possible to use standard tools from linear algebra to generate and structure such codes.

Definition 7. *A* generator matrix *G for a linear code C is a $k \times n$-matrix for which the rows are a basis of C.*

If G is a generator matrix, then $C = \{\mathbf{a}G \,|\, \mathbf{a} \in \mathbb{F}_q^k\}$. It follows immediately that elementary row operations preserve the code. Any k independent columns of G are called an *information set* of C and the remaining $n-k$ columns are called a *parity-check set.*

We say that G is in *standard form* if G is of the form $G=(I_k\;\, P)$, where I_k is a $k\times k$ identity matrix. If a particular instance of the code C is generated using a matrix in standard form, then it is called *systematic.*

Two linear codes are called *equivalent* if they differ at most by a permutation of columns in a generator matrix. It is clear that any linear code is equivalent to some linear code in standard form. We would like to note that some authors also include field automorphisms in the definition of equivalence. Under that definition, two linear codes C and D are equivalent if there is a permutation $\pi:[n]\to[n]$ and an automorphism $\sigma:\mathbb{F}_q\to\mathbb{F}_q$ such that

$$(c_1,\ldots,c_n)\in C \iff (\sigma(c_{\pi(1)})\ldots\sigma(c_{\pi(n)}))\in D.$$

Clearly, if q is prime then the two definitions agree.

In general, if M is the total number of codewords in code C, then one needs to check $\binom{M}{2}$ pairs of codewords to determine the minimum distance $d_{\min}$. However, for linear codes, it is sufficient to check only M words, as shown in the following theorem.

Theorem 1. *For a linear code C, the minimum distance is equal to the minimum weight.*

Proof. For all $\mathbf{x},\mathbf{y}\in C$, the following holds

$$d(\mathbf{x},\mathbf{y})=d(\mathbf{x}-\mathbf{y},\mathbf{0})=w(\mathbf{x}-\mathbf{y}),\text{ and if }\mathbf{x}\in C,\mathbf{y}\in C,\text{ then }\mathbf{x}-\mathbf{y}\in C.$$

Hence, $d_{\min}=\min\limits_{\mathbf{x},\mathbf{y}\in C}d(\mathbf{x},\mathbf{y})=\min\limits_{\mathbf{x}\in C}w(\mathbf{x},\mathbf{y})=w_{\min}$. □

Definition 8. *The dual code $C^\perp$ of an $[n,k]$ code C is defined by*

$$C^\perp=\{\mathbf{y}\in\mathbb{F}_q^n\,|\,\forall\mathbf{x}\in C\;\langle\mathbf{x},\mathbf{y}\rangle=0\}$$

Let $G=(I_k\;\, P)$ be a generator matrix for code C, then H, a generator matrix for the dual code $C^\perp$ has the form $H=(-P^T\;\, I_{n-k})$. This follows from

$$\mathbf{x}\in C \iff \mathbf{x}H^T=\mathbf{0}. \tag{2.1}$$

By convention, H is called a *parity check* matrix of C.

Definition 9. *If C is a linear code with parity check matrix H then for every $\mathbf{x} \in \mathbb{F}_q^n$, we call $\mathbf{x}H^T$ the* syndrome *of $\mathbf{x}$.*

It follows from (2.1) that the codewords are characterized by syndrome $\mathbf{0}$. More generally, since C is a subgroup of $\mathbb{F}_q^n$, we can partition $\mathbb{F}_q^n$ into cosets of C. Then $\mathbf{x}, \mathbf{y}$ belong to the same coset if and only if they have the same syndrome ($\mathbf{x}H^T = \mathbf{y}H^T \iff \mathbf{x} - \mathbf{y} \in C$). Therefore, if a vector $\mathbf{x}$ is received with error pattern $\mathbf{e}$, then $\mathbf{x}$ and $\mathbf{e}$ have the same syndrome. It follows that for minimum distance decoding, one needs to identify a vector $\mathbf{e}$ of minimimum weight that belongs to the same coset as $\mathbf{x}$ and then decode $\mathbf{x}$ as $\mathbf{x} - \mathbf{e}$. The vector $\mathbf{e}$ is the *coset leader* (and might not be unique).

To see how this idea works in practice, consider the following $[6, 3]$ binary code C with a generator matrix

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

and the corresponding parity check matrix

$$H = \begin{bmatrix} 0 & -1 & -1 & 1 & 0 & 0 \\ -1 & 0 & -1 & 0 & 1 & 0 \\ -1 & -1 & 0 & 0 & 0 & 1 \end{bmatrix} \cong \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Let $\mathbf{a} \in \mathbb{F}_2^3$ be the message that the sender wants to send to the receiver. Then $\mathbf{c} = \mathbf{a}G \in C$ is the corresponding codeword. Assume that due to the noise in the communication channel, the receiver gets $\mathbf{x} \in \mathbb{F}_2^6$ instead. Then $\mathbf{e} = (e_1, \dots e_6) = \mathbf{x} - \mathbf{c}$ is the error pattern. In this example,

$$\begin{aligned} e_2 + e_3 + e_4 &= x_2 + x_3 + x_4 := s_1, \\ e_1 + e_3 + e_5 &= x_1 + x_3 + x_5 := s_2, \\ e_1 + e_2 + e_6 &= x_1 + x_2 + x_6 := s_3. \end{aligned}$$

Since the receiver knows $\mathbf{x}$, they also know s_1, s_2, s_3. Hence, in order to identify $\mathbf{c}$, the receiver must determine the most likely error pattern $\mathbf{e}$ given s_1, s_2, s_3. By convention, the most likely pattern is the one with fewest errors, *i.e.*, the smallest number of 1s. It can be easily checked that if $(s_1, s_2, s_3) \neq (1, 1, 1)$ there is a unique choice for $\mathbf{e}$. For example, if $(s_1, s_2, s_3) = (1, 1, 0)$, then the most likely pattern is $(0, 0, 1, 0, 0, 0)$. Other possible error patterns are

$$\{(1, 1, 0, 0, 0, 0), (1, 0, 0, 1, 0, 0), (0, 1, 0, 0, 1, 0), (0, 0, 0, 1, 1, 0), (1, 1, 1, 1, 1, 0)\}$$

but they all have more errors.

If $(s_1, s_2, s_3) = (1, 1, 1)$ the decoder must choose $\mathbf{e}$ from

$$\{(1, 0, 0, 1, 0, 0), (0, 1, 0, 0, 1, 0), (0, 0, 1, 0, 0, 1)\}.$$

Assume that the receiver knows that the probability p of receiving a corrupted symbol is p. If we list all possible error patterns and then decode them using maximum likelihood decoding, we see that all patterns with at most one error are decoded correctly and among other patterns, there exists one with two errors that is also decoded correctly. Hence, the probability of decoding correctly is:

$$r^6 + 6r^5p + r^4p^2,$$

where $r = 1 - p$.

One of the benefits of this algebraic approach is that it decreases the space of alternatives that need to be considered for decoding a message. In general, a linear code C contains q^k codewords and the receiver can get q^n messages. If the code had no structure, then for every message, the receiver would need to compare the received string with all q^k codewords. However, in linear codes, it is enough to consider q^{n-k} coset leaders and if we assume that the rate of the code is relatively high, *i.e.*, $k > n - k$, then $q^{n-k} < q^k$.

There is a useful relationship between the weight of a codeword in C and a parity check matrix H.

Theorem 2. *Let C be a linear code with a parity check matrix H and $\mathbf{x}$ a non-zero codeword in C. Then the columns of H corresponding to* $\mathrm{supp}(\mathbf{x})$ *are linearly dependent. Conversely, if a linear dependence relation with nonzero coefficients exists among S columns of H, then there is a codeword $\mathbf{x}$ in C with* $\mathrm{supp}(\mathbf{x}) = S$.

Proof. By (2.1), we know that $\mathbf{x}H^T = \mathbf{0}$. Since $\mathbf{x}$ is a non-zero codeword, the columns in $\mathrm{supp}(\mathbf{x})$ are linearly dependent. Conversely, if S is a dependent set of columns of H, then there exists a set of coefficients such that $\sum_{i \in S} x_i H_i = 0$, where H_i is a column of H. Then these coefficients define a codeword $\mathbf{x} \in C$ with $\mathrm{supp}(\mathbf{x}) = S$. □

Corollary 1. *A linear code C with a parity check matrix H has a minimum distance d if and only if the cardinality of the smallest minimal dependent set of columns of H is d.*

Next, we prove an important bound for the dimension of linear codes.

Theorem 3. Singleton bound. *Let C be a linear $[n, k, d]$ code. Then any set of $n - d + 1$ coordinates contains an information set and $k \leq n - d + 1$.*

Proof. Let G be a generator matrix of C and S a subset of columns of G with cardinality s and assume that S does not contain an information set. Consider the restriction G_S of G to S. This is a $k \times s$ matrix and $\mathrm{rk}(G_S) < k$ since, by assumption, S does not contain an information set. Hence there exists a non-trivial linear combination of the rows of G_S that equals $\mathbf{0}$ and a codeword $\mathbf{x} \in C$ with 0s on S. Since the rows of G are linearly independent, $\mathbf{x} \neq \mathbf{0}$ and hence $d \leq n - s$ and $s \leq n - d$. In particular, a set S of cardinality $k - 1$ cannot contain an information set, so it follows that $k \leq n - d + 1$. □

Linear codes that satisfy the Singleton bound with equality are called *maximum distance separable (MDS)* codes. MDS codes are usually written more concisely as $[n, k]$ codes with $d = n - k + 1$ implied. Note that repetition codes, *i.e.*, the codes that are generated by $\mathbf{1} = (1, \ldots, 1)$, are $[n, 1]$ MDS codes. An example of a non-repetition MDS code is the code generated by the following generator matrix

$$G = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix};$$

it is a $[4, 3]$ MDS code known as the simple parity check code.

On the other hand, the code generated by

$$G = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix},$$

is not an MDS code, since $\dim C = 2$ but $\{1, 2\}$ is a dependent set.

It is sometimes useful to use the notion of *evaluation* codes [26].

Definition 10. *Let $S = \{P_1, \ldots, P_n\}$ be a set and V an $\mathbb{F}_q$-vector space of maps $\phi : S \to \mathbb{F}_q$. Then the evaluation code $C[V, S, \mathbb{F}_q]$ is the image of the map*

$$i : V \to \mathbb{F}_q^n,\ \phi \mapsto (\phi(P_1), \ldots, \phi(P_n)).$$

Since V is a vector space, all evaluation codes are linear codes. On the other hand, all linear codes are evaluation codes since, if G is a generator matrix, then we can let S be the set of columns of G and V the dual space $(\mathbb{F}_q^k)^\perp$, *i.e.* the $\mathbb{F}_q$-space of all linear forms on $\mathbb{F}_q^k$. For $\mathbf{x}' \in \mathbb{F}_q^k$, we denote the corresponding linear form by ϕ' and the encoding of $\mathbf{x}'$ equals $(\phi'(c_1), \dots, \phi'(c_n))$, where c_i is a column of G.

This view of linear codes is particularly useful when V represents a special class of maps, for example, polynomials of bounded degree. We will discuss such classes of linear codes in the following section.

2.3 Examples of Codes

We shall now describe some of the codes that will be used in this thesis. Here the expositions of Reed–Solomon and Reed–Muller codes follow [12].

2.3.1 Repetition Codes

Recall that the *repetition code* $\mathrm{Rep}[n] \subset \mathbb{F}_q^n$ is a one-dimensional linear vector space generated by $\mathbf{1} = (1, \dots, 1)$. In essence, using $\mathrm{Rep}[n]$ to transmit a message corresponds to retransmitting every message n times.

2.3.2 Reed–Solomon Codes

To construct a Reed–Solomon code, we pick a set of distinct points $S = \{\alpha_1, \dots, \alpha_n\} \subseteq \mathbb{F}_q$. Then a Reed–Solomon code $RS_q[n, k]$ is defined by

$$RS_q[n, k] = \{(p(\alpha_1), p(\alpha_2), \dots, p(\alpha_n)) \in \mathbb{F}_q^n\},$$

where $p \in \mathbb{F}_q[x]$ ranges over all polynomials of degree at most $k - 1$.

The canonical generator matrix of this code is given by

$$G = \begin{bmatrix} 1 & \cdots & 1 \\ \alpha_1 & \cdots & \alpha_n \\ \alpha_1^2 & \cdots & \alpha_n^2 \\ \vdots & \vdots & \vdots \\ \alpha_1^{k-1} & \cdots & \alpha_n^{k-1} \end{bmatrix}.$$

Since any two distinct polynomials of degree less than k agree in at most $k - 1$ points, it means that the minimum distance of a Reed–Solomon code is $n-(k-1)=n-k+1$, so it satisfies the Singleton bound with equality and, hence, is an MDS code.

Clearly, $RS_q[n,1]$ is a repetition code. One can also generalize the concept of Reed–Solomon codes by defining a vector $\mathbf{v}\in(\mathbb{F}_q^*)^n$ of coefficients of the coordinates of the codewords, *i.e.*,

$$GRS_q[n,k,\mathbf{v}]=\{(v_1p(\alpha_1),v_2p(\alpha_2),\ldots,v_np(\alpha_n))\in\mathbb{F}_q^n\}$$

over the same set S of evaluation points as above [10].

Next, we will show that *the dual of a generalized Reed–Solomon code is a generalized Reed–Solomon code.* The proof follows [20] and [13]. We start with two lemmas.

Lemma 1. *For every non-zero $\alpha\in\mathbb{F}_q$, we have $\alpha^{q-1}=1$.*

Proof. Let $u:=\prod_{0\neq\beta\in\mathbb{F}_q}\beta\in\mathbb{F}_q$. Observe that for all non-zero α and β, $\alpha\beta\neq 0$. Then the map $\beta\mapsto\alpha\beta$ is a permutation of the non-zero elements of $\mathbb{F}_q$. Therefore,

$$u=\prod_{0\neq\beta\in\mathbb{F}_q}\beta=\prod_{0\neq\beta\in\mathbb{F}_q}\alpha\beta=\alpha^{q-1}\prod_{0\neq\beta\in\mathbb{F}_q}\beta$$

and since $\alpha\neq 0$, the claim follows. □

Lemma 2. *Let $k\in\mathbb{F}_q$. Then*

$$\sum_{\beta\in\mathbb{F}_q}\beta^k=\begin{cases}0, & \text{if } k\in\mathbb{F}_q\backslash\{q-1\};\\ -1, & \text{if } k=q-1.\end{cases}$$

Proof. First, let $S_k:=\sum_{\beta\in\mathbb{F}_q}\beta^k$. Then, as in Lemma 1, we can apply the permutation $\beta\mapsto\alpha\beta$ with $\alpha\neq 0$ to conclude that

$$S_k=\sum_{\beta\in\mathbb{F}_q}\alpha^k\beta^k=\alpha^k\sum_{\beta\in\mathbb{F}_q}\beta^k=\alpha^kS_k.$$

Now we have two cases: either $S_k=0$, or $S_k\neq 0$ and then $\alpha^k=1$. Assume $S_k\neq 0$. Then the polynomial $x^k-1=0\in\mathbb{F}[x]$ has $q-1$ roots. This can only hold if $k=q-1$. So, we have

$$S_{q-1}=\sum_{\beta\in\mathbb{F}_q}\beta^{q-1}=0^{q-1}+\sum_{0\neq\beta\in\mathbb{F}_q}\beta^{q-1}=q-1,$$

where the final equality follows from Lemma 1. However, $q-1 \equiv -1 \mod q$, and the claim follows. □

Now it is easy to see why

$$G = \begin{bmatrix} 1 & \cdots & 1 \\ \alpha_1 & \cdots & \alpha_n \\ \alpha_1^2 & \cdots & \alpha_n^2 \\ \vdots & \vdots & \vdots \\ \alpha_1^{k-1} & \cdots & \alpha_n^{k-1} \end{bmatrix}$$

and

$$H = \begin{bmatrix} 1 & \cdots & 1 \\ \alpha_1 & \cdots & \alpha_n \\ \alpha_1^2 & \cdots & \alpha_n^2 \\ \vdots & \vdots & \vdots \\ \alpha_1^{n-k-1} & \cdots & \alpha_n^{n-k-1} \end{bmatrix}$$

are two orthogonal matrices.

We will derive the dual of a generalized Reed–Solomon code using Lagrange interpolation. For that, we will need the following notation. Let

$$L(x) := \prod_{i=1}^{n}(x-\alpha_i)$$

and

$$L_i(x) := L(X)/(x-\alpha_i) = \prod_{j\neq i}(x-\alpha_j).$$

Note that $L(x)$ and $L_i(x)$ are monic of degree n and $n-1$, respectively.

First, let us state the famous Lagrange interpolation theorem.

Theorem 4. *Let $p \in \mathbb{F}_q[x]$ and $\deg(p) = d$. Assume that, for distinct $\{\alpha_1, \ldots, \alpha_n\} \subseteq \mathbb{F}_q$ with $d < n$, we have $p(\alpha_i) = \beta_i$. Then*

$$p(x) = \sum_{i=1}^{n} \beta_i (\prod_{j\neq i} \frac{x-\alpha_j}{\alpha_i - \alpha_j}).$$

Proof. Let $g(x)$ be the right-hand side of the equation. Observe that $\deg(g) \leq n-1$ and $g(\alpha_i) = \beta_i$. By assumption, $\deg(p) = d \leq n-1$, so $p(x) - g(x)$ is a polynomial of degree at most $n-1$. But it has n distinct roots, so $p(x) - g(x)$ is the zero polynomial. Hence, $p(x) = g(x)$. □

We are finally ready to determine the dual of a $GRS_q[n, k, \mathbf{v}]$ code.

Theorem 5. *For any* $GRS_q[n, k, \mathbf{v}]$*,*

$$GRS_q[n, k, \mathbf{v}]^{\perp} = GRS_q[n, n-k, \mathbf{u}],$$

where $\mathbf{u}$ *is given by* $u_i^{-1} = v_i \prod_{j \neq i} (\alpha_i - \alpha_j)$*.*

Proof. By construction, $u_i = v_i^{-1} L_i(\alpha_i)$. We want to show that

$$\forall \mathbf{p} \in GRS_q[n, k, \mathbf{v}], \mathbf{g} \in GRS_q[n, n-k, \mathbf{u}] \quad \mathbf{p} \cdot \mathbf{g} = 0.$$

By definition, the ith coordinate of vector $\mathbf{p}$ is $v_i p(\alpha_i)$ and the polynomial p has a degree at most $k-1$. Similarly, the ith coordinate of vector $\mathbf{g}$ is $v_i^{-1} g(\alpha_i)$ and the polynomial p has a degree at most $n-k-1$.

Therefore, their product $p(x)g(x)$ has a degree at most $n-2$, so the coefficient of x^{n-1} is 0. From Lagrange interpolation we have

$$p(x)g(x) = \sum_{i=1}^{n} p(\alpha_i) g(\alpha_i) \frac{L_i(x)}{L_i(\alpha_i)},$$

Recall that $L_i(x)$ is monic of degree $n-1$. We can now equate the coefficients of x^{n-1} on both sides.

$$\begin{aligned} 0 &= \sum_{i=1}^{n} p(\alpha_i) g(\alpha_i) \frac{1}{L_i(\alpha_i)} \\ &= \sum_{i=1}^{n} (v_i p(\alpha_i)) \left(\frac{v_i^{-1}}{L_i(\alpha_i)} g(\alpha_i) \right) \\ &= \mathbf{p} \cdot \mathbf{g}. \end{aligned}$$

□

2.3.3 Reed–Muller Codes

Another useful generalization of Reed–Solomon codes is that of *Reed–Muller codes* $RM_q[r, m]$ defined by

$$RM_q[r, m] = \{(p(\alpha_1), p(\alpha_2), \ldots, p(\alpha_n)) \in \mathbb{F}_q^n\},$$

where the set $\{\alpha_1, \dots, \alpha_n\}$ equals $\mathbb{F}_q^m$, $p \in \mathbb{F}_q[X_1, \dots, X_m]$ is a polynomial of degree at most r, and $n = q^m$.

Hence, the dimension of the code is given by the number of independent polynomials of degree at most r. If $q = 2$, then the dimension k of the code is given by

$$k = 1 + \binom{m}{1} + \cdots + \binom{m}{r}.$$

We want to show how to construct a (non-systematic) generator matrix G of $RM_2[r, m]$. Let us start by introducing some notation.

Definition 11. *The* star (or Schur) *product of two vectors* $\mathbf{a}$ *and* $\mathbf{b}$ *is their componentwise multiplication,* i.e.,

$$\mathbf{a} \star \mathbf{b} = (a_1, \dots, a_n) \star (b_1, \dots, b_n) = (a_1 b_1, \dots, a_n b_n).$$

Then the following block matrix is a generator matrix G for a Reed–Muller code $RM_2[r, m]$:

$$G = \begin{bmatrix} G_0 \\ G_1 \\ \vdots \\ G_r \end{bmatrix},$$

where G_0 is an all-ones vector of length 2^m, G_1 is an $m \times 2^m$ matrix with each binary m-tuple appearing once as a column and G_l is an $\binom{m}{l} \times 2^m$ matrix consisting of all possible l-element star products of rows of G_1. As an example, let us construct a generator matrix for $RM_2[3, 4]$:

$$G_0 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix},$$

$$G_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix},$$

$$G_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix},$$

$$G_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

where we arrange the columns of G_1 from all zeros to all ones in increasing order with the low-order bit in the bottom row.

It can be shown that $RM_2[r, m]$ has minimum distance 2^{m-r} and its dual is $RM_2[m-r-1, m]$[31].

2.4 Standard Operations on Linear Codes

Unless otherwise specified, we follow the notation introduced in [1].

2.4.1 Operations on a Single Code

Let $C[n, k, d]$ be a linear code. One can *extend* the code by fixing k and adding parity check symbols (so $n-k$ and n increase). Note that this operation does not affect the number of codewords in the code. Also, the extension is not unique and d may or may not increase.

Another useful operation is that of *puncturing*. Let $S \subset [n]$ be any set of coordinates and G a generator matrix of C. We produce a punctured code C^S by removing the columns in S from G. By convention, we do not change the index of the columns after puncturing, *e.g.*, if we have 5 coordinates and puncture in the second, then the remaining coordinates are $\{1, 3, 4, 5\}$. Note that if every column in S corresponds to a parity check symbol, then the total number of codewords remains unchanged. Here we generalize the definition of puncturing given in [31] by allowing the removal of information symbols.

Closely related to puncturing is the operation of *shortening*. Consider the same subset S as before, but now construct a shortened code C_S be selecting only those codewords in C whose restriction $\mathbf{x}_{|S} = \mathbf{0}$ and puncturing them on S. We summarise several useful properties of punctured and shortened codes in the following theorem.

Theorem 6. *Let $C[n, k, d]$ be a linear code over $\mathbb{F}_q$ and S a set of s coordinates. Then:*

1. $(C^\perp)_S = (C^S)^\perp$ *and* $(C^\perp)^S = (C_S)^\perp$;
2. *if* $s < d$, *then* $\dim C^S = k$ *and* $\dim(C^\perp)_S = n - s - k$;
3. *if* $s = d$ *and* S *is the support of a minimum weight codeword, then* $\dim C^S = k - 1$ *and* $\dim(C^\perp)_S = n - s - k + 1$.

Proof.

1. Let $\mathbf{x}$ be a codeword of $C^{\perp}$ such that its restriction $\mathbf{x}_{|S} = \mathbf{0}$. We puncture $\mathbf{x}$ in the coordinates in S and denote the resulting codeword by $\mathbf{x}'$. It clear that $\mathbf{x}' \in (C^{\perp})_S$. Now take any $\mathbf{y} \in C$. By construction, $\mathbf{0} = \mathbf{y}\cdot\mathbf{x} = \mathbf{y}'\cdot\mathbf{x}'$, where $\mathbf{y}'$ is the codeword $\mathbf{y}$ punctured in S. Therefore, $\mathbf{x}' \in (C^S)^{\perp}$.

   On the other hand, let $\mathbf{x}$ be a codeword in $(C^S)^{\perp}$. We extend it by adding $\mathbf{0}$ on the coordinates in S and denote the resulting code by $\mathbf{x}''$. Take any $\mathbf{y} \in C$ and puncture it on S to obtain $\mathbf{y}'$. Then $\mathbf{0} = \mathbf{y}' \cdot \mathbf{x} = \mathbf{y} \cdot \mathbf{x}''$,and so $\mathbf{x} \in (C^{\perp})_S$.

2. Assume $s < d$. Then $n-d+1 \leq n-s$, so by Theorem 3, $n-s$ contains an information set. Therefore, $\dim C^S = k$ and the claim follows from part 1.

3. Let $S' \subsetneq S$ have cardinality $|S'| = d-1$. Then by part 2, $\dim C^{S'} = k$ and the minimum distance of $C^{S'}$ is 1. We obtain C^S by puncturing $C^{S'}$ in the only non-zero coordinate of a minimum weight codeword of $C^{S'}$. This means that there is no linear dependency between the elements of S' and $S - S'$, so $\dim C^S = k - |S - S'| = k - 1$.

□

2.4.2 Operations on Collections of Codes

Let $\{C_i[n_i, k_i, d_i], i \in [m]\}$ be a collection of linear codes over field $\mathbb{F}_q$.

The *direct sum* of $\{C_i[n_i, k_i, d_i], i \in [m]\}$ is the code C,

$$C = \bigoplus_i C_i = \{(c_1, \ldots, c_m) | c_i \in C_i\}$$

Note that C is a $[\sum_i n_i, \sum_i k_i, \min\{d_i\}]$ code. The first two parameters are immediately clear from the definition. To see why minimum distance is $\min\{d_i\}$, we apply Theorem 1. Let c_i' be a codeword of minimum weight in C_i. It is clear that to find codewords of minimum length in C, it is enough to compare the codewords of the form $(0, \ldots, c_i', \ldots, 0)$, *i.e.* $c_j = 0$ if $i \neq j$ and $c_i = c_i'$. It follows that the minimum weight w of C is $w = \min\{w_i\} = \min\{d_i\}$. Let G_i be the generator matrix for C_i. Then G, the generator matrix for C becomes

$$G = \bigoplus_i G_i = \begin{bmatrix} G_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & G_m \end{bmatrix}.$$

Similarly, the parity check matrix H is

$$H = \bigoplus_i H_i = \begin{bmatrix} H_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & H_m \end{bmatrix}.$$

Another common operation is called the $(\mathbf{u}, \mathbf{u}+\mathbf{v})$ *construction* or *Plotkin sum.* We assume that $n_1 = n_2 = \cdots = n_m$ and define

$$C := \{(u_1, u_1 + u_2, \cdots, u_1 + u_2 + \cdots + u_m) | u_i \in C_i\}.$$

C is a $[mn, \sum_i k_i, \min\{(m - i + 1)d_i\}]$ code. The fact that the block length is mn is clear from definition. To see that the expressions for dimension and minimum distance hold, note that if $\{u_1, \ldots, u_{k_i}\}$ is a basis for C_i, then $\{(u_1, \ldots, u_1), (0, u_2, \ldots, u_2), \ldots, (0, \ldots, 0, u_m)\}$ is the corresponding basis for C.

Also, the generator matrix G and the parity check matrix H are:

$$G = \begin{bmatrix} G_1 & G_1 & G_1 & \vdots & G_1 \\ 0 & G_2 & G_2 & \vdots & G_2 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & G_i & \vdots & G_i \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \vdots & G_m \end{bmatrix},$$

$$H = \begin{bmatrix} H_1 & 0 & 0 & \vdots & 0 \\ -H_2 & H_2 & 0 & \vdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ -H_i & -H_i & H_i & \vdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ -H_m & -H_m & -H_m & \vdots & H_m \end{bmatrix}.$$

Finally, we consider one more operation called a *star (or Schur) product* of codes as defined in [10]. Let $A[n, k_1, d_1]$ and $B[n, k_2, d_2]$ be two linear codes over the same field. With a minor abuse of notation, we pick a generator matrix for each of the codes and denote them by A and B respectively. Then the star product $A \star B$ is the linear code generated by k_1k_2 vectors corresponding to the star products of rows of the generator matrices A and B, where a star product of two vectors is their component-wise multiplication as given in Definition 11. Clearly, $\dim A \star B \leq k_1k_2$.

Note that this definition is independent of the choice of generator matrices because multiplication is distributive over addition. Furthermore, the star product of codes can be alternatively defined as $\langle a \star b : a \in A[n, k_1, d_1], b \in B[n, k_2, d_2]\rangle$. We refer the reader to [19], [22], [23] for a detailed discussion of the known algebraic properties of star products of linear codes.

2.5 Entropy and Mutual Information

A natural question that comes up during the discussion of encoding is how many bits are really necessary to encode an information symbol. It was addressed already in 1948 by Shannon [25], who established an important lower bound and introduced the concept of entropy.

Definition 12. *Given a discrete random variable X with possible values $\{x_1, \dots, x_n\}$, and probability mass function $P(X)$, the* entropy*, denoted $H(X)$, is given by*

$$H(X) = \mathrm{E}[-\log(P(X))] = -\sum_{i=1}^{n} p_i \log p_i. \tag{2.2}$$

Intuitively, entropy captures the level of uncertainty of a random variable. The base of the logarithm represents the choice of unit of measure. Common choices are 2 (in which case entropy is measured in bits), e (nats), 10 (bans). In our case, we will work with logarithms of base q, where q is the size of our ground field. Let us illustrate this with the following example.

Example 2.5.1. *Let X be a random variable representing the number that comes up on a fair 6-sided die,* i.e.*, $p_i = \frac{1}{6}$ for all i. Then*

$$H(X) = -\sum_{i=1}^{6} \frac{1}{6} \log \frac{1}{6} = 0.78.$$

On the other hand, let Y be a random variable representing the number that comes up on an unfair 6-sided die with $p_1 = \frac{1}{2}$ and $p_i = \frac{1}{10}$ for all $i > 1$. Then

$$H(Y) = -\frac{1}{2}\log\frac{1}{2} - \sum_{i=2}^{6}\frac{1}{10}\log\frac{1}{10} = 0.65.$$

We can see that the $H(X) > H(Y)$, which reflects the fact that X has a higher level of uncertainty.

Entropy was constructed to reflect the intuitive notion of the uncertainty of the random variable and to satisfy three key properties:

1. H should be continuous in each p_i;
2. If all the p_i are equal, *i.e.*, $p_i = \frac{1}{n}$, then H should be a monotonic increasing function of n. This is because if every event is equally likely then the uncertainty should grow in the number of events.
3. If an event be broken down into two successive events, the original H should be the weighted sum of the individual values of H. This means that for positive integers $b_1, \ldots, b_k$ such that $b_1 + \cdots + b_k = n$,

   $$\mathrm{H}_n\left(\frac{1}{n}, \ldots, \frac{1}{n}\right) = \mathrm{H}_k\left(\frac{b_1}{n}, \ldots, \frac{b_k}{n}\right) + \sum_{i=1}^{k}\frac{b_i}{n}\mathrm{H}_{b_i}\left(\frac{1}{b_i}, \ldots, \frac{1}{b_i}\right),$$

   where subscripts are used to indicate the number of outcomes in each event.

Let us illustrate the last condition with an example.

Example 2.5.2. *Let X be a random variable representing the choice between three options with probabilities $p_1 = \frac{1}{2}, p_1 = \frac{1}{3}, p_1 = \frac{1}{6}$. This random variable can be interpreted as two successive choices: first, choosing between the first option or the remaining options, both with probability $\frac{1}{2}$ and, second, if options $\{2, 3\}$ are chosen, then choosing 2 with probability $\frac{2}{3}$ and 3 with probability $\frac{1}{3}$. We have*

$$H\left(\frac{1}{2}, \frac{1}{3}, \frac{1}{6}\right) = H\left(\frac{1}{2}, \frac{1}{2}\right) + \frac{1}{2}H\left(\frac{2}{3}, \frac{1}{3}\right).$$

Shannon showed that, up to a constant coefficient, the only function that satisfies all three requirements is the one given in 2.2. The constant coefficient is used for convenience and reflects the choice of unit of measure, *i.e.*, the choice of the base of logarithms.

Observe, that from the perspective of the receiver, the symbols transmitted by the sender are a random variable. Using this perspective and entropy, Shannon showed that

1. The average number of bits per symbol of any uniquely decodable source must be greater than or equal to the entropy H of the source;
2. If the string of symbols is sufficiently large, there exists a uniquely decodable code for the source such that the average number of bits per symbol of the code is arbitrarily close to H.

The concept of entropy can be extended to several random variables.

Definition 13. *Given two discrete random variables X and Y with joint probability mass function $P(X, Y)$, the* joint entropy*, denoted by $H(X, Y)$ is given by*

$$H(X, Y) = \mathrm{E}[-\log(P(X, Y))] = -\sum_{x,y} p(x, y) \log p(x, y).$$

Joint entropy measures the uncertainty when the two random variables X and Y are taken together.

Definition 14. *Given two discrete random variables X and Y with joint probability mass function $P(X, Y)$ and conditional probability mass function $P(X|Y)$, the* conditional entropy *of X given Y, denoted by $H(X|Y)$ is given by*

$$H(X|Y) = \mathrm{E}[-\log(P(X|Y))] = -\sum_{x,y} p(x|y) \log p(x|y).$$

Entropy has the following properties:

- Entropy is non-negative and $H(X) = 0$ if and only if X is deterministic.
- Joint entropy can be decomposed as follows:

  $$H(X_1, \ldots, X_n) = \sum_{i=1}^{n} H(X_i|X_1, \ldots, X_{i-1}).$$

  In particular,

  $$H(X, Y) = H(X|Y) + H(Y) = H(Y|X) + H(X).$$

- Conditioning decreases entropy

  $$H(X|Y) \leq H(X)$$

- Let χ be the set of outcomes of X. Then
$$H(X) \leq \log(|\chi|).$$
- Entropy is maximized and the bound from the previous point is achieved when X is uniformly distributed.
- Entropy is non-increasing under functions:
$$H(X) \geq H(g(X)),$$
where $g(X)$ is a deterministic function of X. The equality is achieved if and only if g is a bijection on a set of probability 1.

As we have seen above, entropy describes the level of uncertainty of a random variable X and conditional entropy shows how much uncertainty about X remains once we know the value of another random variable Y. So, a natural next step would be to connect these two concepts and see, how much uncertainty in X cannot be related to Y. This is where mutual information comes in [32].

Definition 15. *The* mutual information *between two discreet random variables X and Y jointly distributed according to $p(x, y)$ is given by*

$$\mathrm{I}(X, Y) = \sum_{x,y} p(x, y) \log \frac{p(x, y)}{p(x)p(y)}.$$

It follows immediately from the definition that mutual information is symmetric and non-negative, and, using the properties of entropy, we see that

$$\begin{aligned}\mathrm{I}(X, Y) &= \mathrm{H}(X) - \mathrm{H}(X|Y)\\ &= \mathrm{H}(Y) - \mathrm{H}(Y|X)\\ &= \mathrm{H}(X) + \mathrm{H}(Y) - \mathrm{H}(X, Y).\end{aligned}$$

Observe that mutual information $\mathrm{I}(X, Y)$ is zero if and only if the random variables X and Y are independent.

In the context of private information retrieval, the two random variables of interest are the queries that are received by servers and the identity of the file of interest. A priori, servers do not know which queries they are going to receive but they know their distribution, so queries are a random variable from their perspective. Similarly, the identity of the file of interest is unknown to them but they know the entire set of files and the distribution of interests among the user population in general.

Chapter 3

Combinatorial Preliminaries

There are two combinatorial concepts that are going to be heavily used in our work. The first one is the abstract simplicial complex, which is the combinatorial analog of a topological simplicial complex. This is going to be how we think about collusion patterns. The second one is matroid, which is going to describe the combinatorial properties of a linear code. We will discuss both in this chapter.

3.1 Abstract Simplicial Complex

We will summarise some facts about abstract simplicial complexes in this section and refer the reader to [21], [18], [33].

Definition 16. *An* abstract simplicial complex *is a collection $\mathcal{S}$ of finite nonempty sets such that if S is an element of $\mathcal{S}$, then so is every nonempty subset of S.*

An element $S \in \mathcal{S}$ is called a simplex or face of $\mathcal{S}$. An inclusion maximal element of $\mathcal{S}$ is called a facet. We say that $\mathcal{S}$ is generated by its facets if

$$\mathcal{S} = \langle S_1, \ldots, S_n \rangle := \{S \subseteq S_i \quad \forall i \in [n]\}.$$

There is also the notion of dimension.

$$\begin{aligned} \dim S &= |S| - 1 \\ \dim \mathcal{S} &= \begin{cases} \max\limits_{S \in \mathcal{S}} \dim S, & \text{if this maximum is finite;} \\ \infty, & \text{otherwise.} \end{cases} \end{aligned}$$

A simplicial complex is called pure if all facets have the same dimension.

The vertex set of $\mathcal{S}$ is the union of singleton faces of $\mathcal{S}$ and is denoted by $V(\mathcal{S})$. Clearly, an element is a vertex $v \in V(\mathcal{S})$ if and only if it is also a 0-simplex of $\mathcal{S}$.

An abstract simplicial complex if called finite if it has finitely many faces, or equivalently if its vertex set is finite. A subcollection of $\mathcal{S}$ that is also an abstract simplicial complex is called a subcomplex of $\mathcal{S}$.

Two abstract simplicial complexes $\mathcal{S}$ and $\mathcal{S}'$ are called isomorphic if there exists a bijection $f : V(\mathcal{S}) \to V(\mathcal{S}')$ such that $\{v_0, \ldots, v_k\} \in \mathcal{S}$ if and only if $\{f(v_0), \ldots, f(v_k)\} \in \mathcal{S}'$.

It is possible to associate to every abstract simplicial complex $\mathcal{S}$ a piecewise-linear topological space K, which is called the geometric realization of $\mathcal{S}$. It can be shown that every abstract simplicial complex $\mathcal{S}$ of dimension d has a geometric realization of K in $\mathbb{R}^{2d+1}$ and the realization is unique up to a linear isomorphism.

3.2 Matroids and Coding Theory

We will now define the concept of *matroids* that will be used in later sections to analyze the combinatorics of retrieval codes. Our exposition is based on [11], [34], [15]. Matroids generalize the concept of linear independence in vector spaces. There are many equivalent ways to define a matroid, but for the purposes of this thesis, the most useful one is the following.

A matroid M is a pair $(E, \mathcal{C})$, where E is a finite set and $\mathcal{C} \subseteq 2^E$ is a collection of subsets of E, called *circuits*, that satisfies the following axioms:

1. $\emptyset \notin \mathcal{C}$
2. If $C', C'' \in \mathcal{C}$ and $C' \subseteq C''$, then $C' = C''$
3. If $C', C'' \in \mathcal{C}$, $C' \neq C''$ and $e \in C' \cap C''$, then $\exists C \in \mathcal{C}$ such that $C \subseteq (C' \cup C'') - e$.

The terminology arises from graph theory and the circuits of a matroid can be understood as cycles of a graph on edge set E. The circuits of a matroid are the minimal dependent sets, *i.e.*, every proper subset of a circuit is said to be an independent set. We say that a set is dependent if it contains a circuit and it is independent if it is not dependent.

An equivalent definition of a matroid M via its set $\mathcal{I}$ of independent subsets of the ground set E is given by the following axioms:

1. $\emptyset \in \mathcal{I}$
2. If $I \in \mathcal{I}$ and $I' \subseteq I$, then $I' \in \mathcal{I}$
3. If $I', I'' \in \mathcal{I}$ and $|I'| < |I''|$, then $\exists e \in I'' - I'$ such that $I_1 \cup e \in \mathcal{I}$.

In general, any matrix has a corresponding matroid where the columns are the ground set of the matroid and the dependencies of the matroid match precisely the column dependencies of the matrix. In particular, in the case of a linear code D, the matroid of D is given by its generator matrices where E is the set of columns of any such matrix and circuits are given by the supports of minimal linear equations. For example, if D is an $[n, k]$ MDS code, then the corresponding matroid is called *uniform* $U_{k,n}$ and the set of circuits is the collection of all sets of coordinates of cardinality $k + 1$.

Not all matroids have a corresponding matrix, but in our work, we are focusing on $\mathbb{F}_q$-linear matroids, *i.e.*, the matroids that have a representation as a vector space over a field $\mathbb{F}_q$.

For any $S \subseteq E$ we can define the rank $r(S)$ as the size of the largest independent set contained in S. In particular, $r(E)$ of the matroid of an $\mathbb{F}_q$-linear code D is equal to the dimension of the code.

We will now define several useful operations on matroids.

Definition 17. *Let $M = (E, \mathcal{I})$ be a matroid and $X \subseteq E$.*

- *The deletion of M by X is the matroid $M\backslash X = (E - X, \mathcal{I}\backslash X)$, where $\mathcal{I}\backslash X := \{I \in \mathcal{I} : I \subseteq E - X\}$. We will also give an alternative definition in terms of circuits: $M\backslash X = (E - X, \mathcal{C}\backslash X)$, where $\mathcal{C}\backslash X := \{C \in \mathcal{C} : C \subseteq E - X\}$.*
- *The contraction of M by X is the matroid $M/X = (E-X, \mathcal{I}/X)$, where $\mathcal{I}/X := \{I \in \mathcal{I} : I \subseteq E - X$ and there exists a maximal independent set $J \subseteq X$ s.t.$I \cup J \in \mathcal{I}\}$. Alternatively, $\mathcal{C}/X := \{S \in 2^E : \exists S' \subseteq X$ s.t. $S \cup S' \in \mathcal{C}\}$.*

It can be shown that deletion corresponds to puncturing the linear code by columns in X, and contraction corresponds to shortening the linear code by columns in X [6].

A *minor* of a matroid M is a matroid that is obtained from M by a sequence of deletions and contractions. It can be shown that this sequence is associative and that the operations of contraction and deletion commute [34]. This implies that every minor can be written $M\backslash Y/X$ for $X, Y \subseteq E$ and $X \cap Y = \emptyset$.

Similar to the notion of the direct sum of codes, we can define the *direct sum* of matroids [6].

Definition 18. *Let $M_1 = (E_1, \mathcal{I}_1), \ldots M_m = (E_m, \mathcal{I}_m)$ be a collection of m matroids on disjoint ground sets $E_1, \ldots E_m$. Then the direct sum $\bigoplus_i M_i(E_i, \mathcal{I}_i)$ is the matroid $M(\bigcup_{i\in[m]} E_i, \mathcal{I})$, where $\mathcal{I} := \{\bigcup_{i\in[m]} X_i : X_i \in \mathcal{I}_i\}$.*

Chapter 4

Private Information Retrieval

Recall that private information retrieval is a collection of methods that allow the users of a distributed storage system to access desired files without revealing the identities of the files of interest. The concept was first introduced by Chor, Goldreich, Kushilevitz, and Sudan in [7], [8] and extended to encoded distributed storage systems by Shah, Rashmi, Ramchandran, and Kumar [24]. Note that in this literature, the collections of files stored on distributed storage systems are referred to as databases.

In the classic PIR model of [8], a database is a binary m-bit string $x = [x_1, \dots, x_m] \in \{0,1\}^m$ and the user wants to retrieve the ith bit x_i without revealing the index i. Since then, the approach has been extended and we will follow the convention by Freij-Hollanti, Gnilke, Hollanti and Karpuk [10], where the distributed storage system contains m files and each file is a $b \times n$ matrix whose entries are elements of a finite field $\mathbb{F}_q$.

The study of PIR has two flavours: information-theoretic or computational. *Information-theoretic* PIR was introduced in [7] and requires that queries do not contain any file-identifying information. *Computational* PIR was introduced shortly afterward by Kushilevitz and Ostrovsky in [16] with the goal to make it computationally hard to identify the file.

We have said that PIR protects the identities of the files of interest, but we have not specified the adversaries whose presence necessitates such measures. Traditionally, the main type of adversary is thought to be the operator of the distributed storage system who is curious about the exact nature of the queries [7], [8]. Another possible scenario is when a distributed storage system is known to have been hacked by third parties who can now observe queries but the users want to be able to continue downloading files. In the earliest studies of PIR, it has been assumed that either the operator collects query data from all servers simultaneously or then the data from distinct servers cannot be collated. When all servers collude, the only approach that completely guarantees the privacy of users is to download all files. When none of the servers collude, it has been shown that more efficient schemes exist even when there are only two servers in the storage system [8]. Since these two situations present two opposite ends of the spectrum, it is natural to consider what happens in the space between the two extremes. For example, if the distributed storage system is encoded using an $[n, k]$ MDS storage code and the servers can be partitioned into two disjoint groups of colluding servers, each of size at most k, then there exists a retrieval scheme whose rate is $\frac{n-k}{n}$ [30]. Our work contributes to the study of PIR with arbitrary collusion patterns.

The construction of information-theoretic PIR schemes was addressed in [28], [10], [4], [24] and was followed by the study of efficiency of different schemes [2], [27], [3]. One particular question of interest is the derivation of the PIR capacity for different schemes, that is, the highest rate at which a file can be downloaded privately. For example, for the special case of the so-called MDS codes, it has been shown that if a non-colluding coded distributed storage system contains m messages, then the PIR capacity is $C = \frac{1-R}{1-R^m}$, where $R = \frac{k}{n}$ is the rate of the underlying $[n, k]$ MDS storage code [2].

One can deepen the question of developing new schemes by analyzing specific settings for the PIR such as certain important classes of storage codes [9], [17]. On the other hand, it is also possible to generalize the question of PIR by relaxing certain assumptions about the set-up. For example, some servers could be unresponsive or known to provide erroneous information [29].

Our work falls under the category of information-theoretic PIR. We build on the *star-product scheme* that generalizes several previously established schemes [10]. The name comes from component-wise multiplication of vectors also known as the star-product multiplication. In this scheme, queries are codewords chosen uniformly at random from a publicly known linear code, which is referred to as the retrieval code. In order to be able to download the necessary file, the user modifies the queries corresponding to the desired file index in a pre-determined way that cannot be observed by the servers. During the information retrieval process, the queries are star-multiplied with the corresponding vectors in the distributed storage system. Specifically, the servers correspond to the coordinates and each performs one of the coordinate-wise multiplications. The fact that the query modification follows a pre-determined algorithm allows the users to retrieve the original file; while the fact that the modifications are not observable to the servers preserves privacy.

A complete understanding of this scheme will require the knowledge of combinatorics of the interaction of the retrieval code and the collusion pattern and the algebraic properties of the star product of the storage code and the retrieval code. The focus of this work is the combinatorics of the retrieval code and the collusion pattern.

We will use this chapter to introduce the key ideas behind PIR and then move on to present our main results in the subsequent chapters.

4.1 Coded Storage

First, we need to define the concept of an encoded collection of files stored on a distributed storage system. Consider a finite collection of files

$$X = \{x_1, x_2, \dots, x_m\};$$

we assume that the ordering of the files is fixed and publicly known. Each file x_i is an element of $\mathbb{F}_q^{b_i \times k}$, where $\mathbb{F}_q$ is the finite field of size q. In other words, each file x_i can be viewed as a string of symbols that can be split into b_i substrings all of length k. In general, b_i does not need to be the same for all files, but for simplicity, we will assume that it is the case, *i.e.*, $b_i = b$ for all i. Here again, the position of each symbol is fixed and publicly known. Hence, we can represent X as an $mb \times k$ matrix:

$$X = \left[\begin{array}{cccc}
x_{11}^1 & x_{11}^2 & \dots & x_{11}^k \\
x_{12}^1 & x_{12}^2 & \dots & x_{12}^k \\
\dots & \dots & \dots & \dots \\
x_{1b}^1 & x_{1b}^2 & \dots & x_{1b}^k \\
\hline
\dots & \dots & \dots & \dots \\
\hline
x_{i1}^1 & x_{i1}^2 & \dots & x_{i1}^k \\
x_{i2}^1 & x_{i2}^2 & \dots & x_{i2}^k \\
\dots & \dots & \dots & \dots \\
x_{ib}^1 & x_{ib}^2 & \dots & x_{ib}^k \\
\hline
\dots & \dots & \dots & \dots \\
\hline
x_{m1}^1 & x_{m1}^2 & \dots & x_{m1}^k \\
x_{m2}^1 & x_{m2}^2 & \dots & x_{m2}^k \\
\dots & \dots & \dots & \dots \\
x_{mb}^1 & x_{mb}^2 & \dots & x_{mb}^k
\end{array}\right],$$

where a symbol x_{ij}^l is in row j and column l in the matrix of file i.

This collection X is passed to a set E of servers for storage with $|E| = n$. The files need to be protected against data loss in the case of server failure, so each file x_i is encoded using an error-correcting linear $[n, k, d]$ code C (the same for all files) with a fixed and publicly known generator matrix G_C. Note that n equals the total number of servers in the storage system. The encoded files stored on the servers have the form

$$Y = XG_C = \left[\begin{array}{cccc} y_{11}^1 & y_{11}^2 & \cdots & y_{11}^n \\ y_{12}^1 & y_{12}^2 & \cdots & y_{12}^n \\ \ldots & \ldots & \ldots & \ldots \\ y_{1b}^1 & y_{1b}^2 & \cdots & y_{1b}^n \\ \hline \ldots & \ldots & \ldots & \ldots \\ \hline y_{i1}^1 & y_{i1}^2 & \cdots & y_{i1}^n \\ y_{i2}^1 & y_{i2}^2 & \cdots & y_{i2}^n \\ \ldots & \ldots & \ldots & \ldots \\ y_{ib}^1 & y_{ib}^2 & \cdots & y_{ib}^n \\ \hline \ldots & \ldots & \ldots & \ldots \\ \hline y_{m1}^1 & y_{m1}^2 & \cdots & y_{m1}^n \\ y_{m2}^1 & y_{m2}^2 & \cdots & y_{m2}^n \\ \ldots & \ldots & \ldots & \ldots \\ y_{mb}^1 & y_{mb}^2 & \cdots & y_{mb}^n \end{array}\right].$$

We call the set E of servers together with storage code C and data matrix Y a *distributed storage system* [3], [28], [10].

To simplify the exposition, we introduce several conventions. First, the generator matrix G_C is assumed to be in the reduced row echelon form. Second, we assume that $\text{supp}(C) = E$, for if for some server $i \in E$, $c_i = 0$ for all $c \in C$, we don't need to request information from server i. Third, going forward, we will use subscripts to identify files and superscripts to identify servers. For example, y_{ij}^l is the symbol that is stored on server l that comes from encoding row j of file i[1].

4.2 Collusion pattern

Next, we address the question of collusion. In order to reconstruct the identity of the file, one needs to be able to extract this information from the

[1] This is different from the notation introduced in [10], where superscripts denote files and subscripts denote servers.

queries that are sent to one or several servers. We assume that the information extraction is not transitive, *i.e.*, if an adversary (*e.g.*, the operator of the distributed storage system) can access the queries sent to sets T and T' of servers, it does not mean that he or she can collate the information from $T \cup T'$. This can happen when certain groups of servers are placed in different geographical locations. Also, such sets are completely determined by the adversary and our goal is to construct a retrieval scheme that is immune to a given privacy breach.

We say that a set T is a colluding set if the adversary is able to access queries sent to every server in T and we call the collection of such colluding sets a collusion pattern $\mathcal{T}$.

Note that if a set T is a colluding set, then every subset of T is also a colluding set. In other words, we can view collusion patterns as abstract simplicial complexes on the ground set E.

4.3 Privacy of PIR

It is now a good time to formalize what we mean by information-theoretic privacy. Let A^T be the joint distribution of columns in the query matrix Q indexed by the servers in T over all s iterations of the PIR scheme. Recall that we denote the mutual information of two random variables by $I(\cdot;\cdot)$. Then we require that

$$I(A^T; i) = 0 \quad \forall T \in \mathcal{T}.$$

Clearly, under the trivial PIR scheme, where we download all encoded symbols for all files, the privacy is preserved for any $\mathcal{T}$. In particular, this is the only scheme where privacy is preserved when there is only one server or servers share information about the queries among themselves. However, this option is usually prohibitively expensive and we want to develop schemes that download significantly fewer.

4.4 Construction of a PIR Scheme

We are ready to discuss the main building blocks of a PIR scheme for a distributed storage system; this construction is adapted from [10].

Intuitively, a *PIR scheme* has four components:

1. an algorithm for forming queries to retrieve a given file that does not reveal the identity of the file;
2. (server-determined but known to the user) algorithm for how servers compute their response to query;
3. an algorithm for reconstructing the original file from servers' responses;
4. an iteration algorithm for creating new queries for the same file if it is not possible to retrieve all necessary information at once.

All four algorithms are publicly known and only the first component ensures privacy in the scheme. We will now discuss each component in more detail.

4.4.1 Queries

We want to view a query as a block matrix $A \in \mathbb{F}_q^{bm \times n}$, where each block $q_i \in \mathbb{F}_q^{b \times n}$ corresponds to one file in the distributed storage system and the number of blocks corresponds to the total number of files, though some blocks can be zero submatrices. The entry q_{ij}^l is sent to the lth server to retrieve the encoded symbol from the jth row of file i.

Each $q_i \in \mathbb{F}_q^{b \times n}$ is sampled from a publicly known probability space (Q_i, μ_i) and we denote the Cartesian product of all such probability spaces by $\mathcal{X} = Q_1 \times \cdots \times Q_m$. We define the probability space $\mathcal{Q}$ of all possible queries by $\mathcal{Q} := (\mathcal{X}, \nu)$, where ν is the joint distribution of $\{Q_1, \ldots, Q_m\}$. A query A can be also viewed as an element of the m-fold Cartesian product $Q_1 \times \cdots \times Q_m$ where each m-tuple is chosen according to ν.

The process of constructing queries poses the key dilemma of PIR: on the one hand, no query should reveal any information about the identity of the file of interest, on the other hand, we want to minimize the amount of extraneous information that we download.

Exactly how this balance is achieved will become clear later, but at this stage, we would like to say that, in order to retrieve the ith file our query matrix A will have the form

$$A = \left[\begin{array}{cccc}
q_{11}^{1} & q_{11}^{2} & \cdots & q_{11}^{n} \\
q_{12}^{1} & q_{12}^{2} & \cdots & q_{12}^{n} \\
\cdots & \cdots & \cdots & \cdots \\
q_{1b}^{1} & q_{1b}^{2} & \cdots & q_{1b}^{n} \\
\hline
\cdots & \cdots & \cdots & \cdots \\
\hline
q_{i1}^{1} & q_{i1}^{2} & \cdots & q_{i1}^{n} \\
q_{i2}^{1} & q_{i2}^{2} & \cdots & q_{i2}^{n} \\
\cdots & \cdots & \cdots & \cdots \\
q_{ib}^{1} & q_{ib}^{2} & \cdots & q_{ib}^{n} \\
\hline
\cdots & \cdots & \cdots & \cdots \\
\hline
q_{m1}^{1} & q_{m1}^{2} & \cdots & q_{m1}^{n} \\
q_{m2}^{1} & q_{m2}^{2} & \cdots & q_{m2}^{n} \\
\cdots & \cdots & \cdots & \cdots \\
q_{mb}^{1} & q_{mb}^{2} & \cdots & q_{mb}^{n}
\end{array}\right],$$

where for all $j \neq i$, the blocks q_j will be sampled from (Q_j, μ_j); whereas the probability space (Q_i, μ_i) will be replaced with another probability space $(Q_i^{\mathcal{T}}, \mu_i^{\mathcal{T}})$ that is indistinguishable from (Q_i, μ_i) for the collusion pattern $\mathcal{T}$.

We will focus on the queries that are generated according to the following conventions:

- $(Q_i, \mu_i) = (Q_j, \mu_j)$ for all i and j, and so we will suppress the notation and write (Q, μ) instead. It will follow from the construction that under this assumption, $(Q_i^{\mathcal{T}}, \mu_i^{\mathcal{T}}) = (Q_j^{\mathcal{T}}, \mu_j^{\mathcal{T}})$ for all i and j, so we will also use the notation $(Q^{\mathcal{T}}, \mu^{\mathcal{T}})$
- μ and $\mu^{\mathcal{T}}$ are the discrete uniform distributions on Q and $Q^{\mathcal{T}}$, respectively;
- $\nu = \prod_{i=1}^{m} \mu_i$, *i.e.*, the blocks are chosen independently;
- Q is a linear code and we will construct $Q^{\mathcal{T}}$ as a linear code.

Finally, we assume that the uploading cost, *i.e.*, the cost of sending queries, is negligible compared to the downloading cost, which is commonly encountered in practice. For example, it is much cheaper to select a movie rather than to stream one. In our model, this can be guaranteed by letting the queries take values in a small subfield of the field $\mathbb{F}_q$ of the storage code and was explicitly done for the first time in [9].

4.4.2 Responses

The convention in the PIR literature is that servers compute responses by taking inner products of the columns of Y and the columns of Q, *i.e.* $r^j = \langle q^j, y^j \rangle \in \mathbb{F}_q$ (hence, our schemes belong to the class of *linear* PIR schemes). We call $R = [r^1 \cdots r^n]$ *the response vector* which is transmitted to the user.

4.4.3 Reconstruction function

Next, we need a *reconstruction function* which takes as input the response vector R and returns information symbols from the file x^i. The goal of this step is to separate decoy queries from information-retrieving queries. We will be dealing only with strongly linear PIR schemes, *i.e.*, in addition to responses being given by inner products, our reconstruction functions will be linear transformations whose matrix is a generator matrix of the dual of a certain linear code [14]. In other words, the reconstruction function will be a parity check matrix of the code itself. This code will be designed to contain all responses to decoy queries.

4.4.4 Iteration process

A complete reconstruction of row j of file x_i requires the knowledge of all k information symbols. Depending on the encoding and the collusion pattern, it is possible that we can download some $c \neq k$ symbols during a single iteration. This may allow us to improve the scheme by iterating $s > 1$ times if the number of rows per file $b > 1$.

First, observe that since any server's response is an inner product of the respective columns of Y and Q, a given subset of columns can be used for downloading data from at most one row during one iteration. Otherwise, we will only be able to reconstruct the file up to sums of rows. Therefore, in any individual iteration, once we fix a set of servers for a given row of the file, we do not attempt to download symbols that are stored on the same set from other rows.

Second, it is possible that during some iterations, one can download more symbols from a given row than is necessary. This could be because the collusion pattern and the encoding allow downloading $c > k$ symbols per row or that after a certain number of iterations, one needs to download $k' < k$ remaining symbols, but it is possible to download $k' < c < k$ symbols.

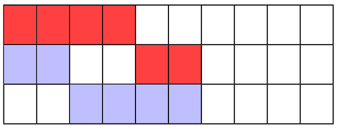

Figure 4.1: Visualizing the iteration process in Example 4.4.1. There are $n = 10$ servers, each file contains $b = 3$ rows with 4 information symbols per row and we can download 6 symbols per iteration. Depicted is the encoded file $y_i \in \mathbb{F}_q^{b \times n}$, along with the encoded symbols downloaded in the first (red) and second (blue) iterations.

Given these two observations, the choice of servers that are accessed during a given iteration is trade-off between maximizing the number of symbols downloaded from one row versus the number of rows accessed.

Example 4.4.1. *Consider the situation when the number of servers* $n = 10$*, each file contains* $b = 3$ *rows, there are* 4 *information symbols per row and we can download* 6 *symbols per iteration (see Fig. 4.1). Naturally, one can download all three rows in three iterations, downloading* 6 *symbols from one row every time, but this will be wasteful. Instead, one could adopt the following strategy:*

1. *During the first iteration, download first-row symbols from servers* $\{1, 2, 3, 4\}$ *and second-row symbols from servers* $\{5, 6\}$*.*
2. *During the second iteration, download second-row symbols from servers* $\{1, 2\}$ *and third-row symbols from servers* $\{3, 4, 5, 6\}$*.*

This (not unique) strategy shows that it is possible to download all three rows in two iterations. Note that since we need a total of 12 *symbols to reconstruct the original file and it is possible to download at most* 10 *symbols per iteration, then the strategies that involve two iterations maximize efficiency. This is illustrated in Figure 4.1.*

4.5 The Class of Star Product Schemes

We will now discuss a class of PIR schemes that arise from the operation of the star product that was discussed in Sec.2.4.2. This class is a generalization of the star product scheme introduced in [10].

Let C be a linear $[n, k, d]_q$ storage code, X a matrix of files, and $Y = XG_C$ a distributed storage system as in Section 4.1. We denote the set of all servers by E with $|E| = n$. Our queries will be generated using a linear code Q, which we call a *retrieval code.*

Before we continue, we would like to point out an important observation that will help understand how the star product schemes protect privacy.

Lemma 3. *Let $\mathcal{A}$ and $\mathcal{B}$ be two linear codes and we construct two $m \times n$ matrices A and B, such that each row of each matrix is a codeword in the respective linear code. We denote rows by subscripts, so $A_i \in \mathcal{A}$ and $B_i \in \mathcal{B}$. If we now take a vector $\mathbf{v} = [v^1 \cdots v^n]$, where $v^j = \langle A^j, B^j \rangle$, and superscripts indicate columns, then $\mathbf{v} \in \mathcal{A} \star \mathcal{B}$.*

Proof. This is because

$$\begin{aligned} \mathbf{v} = [v^1 \cdots v^n] &= [\langle A^1, B^1 \rangle \cdots \langle A^n, B^n \rangle] && (4.1) \\ &= \sum_{i=1}^{m} [A_i^1 \star B_i^1 \cdots A_i^n \star B_i^n] && (4.2) \\ &= \sum_{i=1}^{m} A_i \star B_i \in \mathcal{A} \star \mathcal{B}. && (4.3) \end{aligned}$$

Hence, if every row in the query matrix A is a codeword in Q, then the response vector R is a codeword in $C \star Q$. In particular, R will be annihilated by a generator matrix of $(C \star Q)^{\perp}$. □

We will now introduce some useful notation and a definition.

Let $d(x, y)$ be the Hamming distance of two strings x and y of equal length, and let C, Q be codes. We write:

- $d_{|S}(x, y)$ to denote the Hamming distance of x and y restricted to a subset S of coordinates, *i.e.* $d_{|S}(x, y) = |\{i | x_i \neq y_i, \quad i \in S\}|$;
- $d(x, C) := \min_{y \in C} d(x, y)$ and $d_{|S}(x, C) := \min_{y \in C} d_{|S}(x, y)$;
- $d(C, Q) := \min_{x \in C, y \in Q} d(x, y)$ and $d_{|S}(C, Q) := \min_{x \in C, y \in Q} d_{|S}(x, y)$.

Definition 19. *Let $\mathcal{T}$ be a collusion pattern and Q a retrieval code on the same ground set, and let $T \in \mathcal{T}$. We say that Q^T is a* lift *of Q over the* colluding set *T if $Q^T = \{\mathbf{x} \in \mathbb{F}_q^n : d_{|T}(\mathbf{x}, Q) = 0\}$. We denote the elements of Q^T by $\mathbf{q}^T$.*

Similarly, we say that $Q^{\mathcal{T}}$ is a lift *of Q over the* ***collusion pattern*** *$\mathcal{T}$ if $Q^{\mathcal{T}} = \{\mathbf{x} \in \mathbb{F}_q^n : d_{|T}(\mathbf{x}, Q) = 0 \quad \forall T \in \mathcal{T}\}$, and we denote the elements of $Q^{\mathcal{T}}$ by $\mathbf{q}^{\mathcal{T}}$.*

Note that our definion allows for arbitrary colluding sets. This is different from [10], where the authors focused on t-collusion, *i.e.*, when every set of size at most t is a colluding set.

It is immediate from the definition that $Q \subseteq Q^T$ for all $T \in \mathcal{T}$ and $Q^{\mathcal{T}} = \cap_{T \in \mathcal{T}} Q^T$. Each Q^T is a linear code, for if $\mathbf{q}^T$ and $\mathbf{q}'^T$ are in Q^T, then for all non-zero $\alpha \in \mathbb{F}_q$,

$$\begin{aligned} d_{|T}(\alpha\mathbf{q}^T + \mathbf{q}'^T, Q) &\leq d_{|T}(\alpha\mathbf{q}^T, Q) + d_{|T}(\mathbf{q}'^T, Q) \\ &= d_{|T}(\mathbf{q}^T, Q) + d_{|T}(\mathbf{q}'^T, Q) \text{ since } \alpha \neq 0 \\ &= 0. \end{aligned}$$

Hence, $Q^{\mathcal{T}}$ is an intersection of linear codes, so it is a linear code and $Q \subseteq Q^{\mathcal{T}} \subseteq \mathbb{F}_q^n$. Also, observe that every $\mathbf{q}^{\mathcal{T}} \in Q^{\mathcal{T}} \backslash Q$ can be written as $\mathbf{q}^{\mathcal{T}} = \mathbf{q} + \mathbf{w}$, where $\mathbf{q} \in Q$ and $\mathbf{w} \notin Q$ and $w^j = 0$ whenever $q^{\mathcal{T}^j} = q^j$. However, this decomposition is not unique.

Example 4.5.1. *Let $Q \subseteq \mathbb{F}_2^5$ be the code generated by*

$$G_Q = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \end{bmatrix}$$

and let $\mathcal{T} = \{\{1, 2, 3\}, \{3, 4, 5\}\}$.

Then $Q^{\mathcal{T}} = \mathbb{F}_2^5$, so it is generated by

$$G_Q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

This is because $r(\{1, 2, 3\}) = r(\{3, 4, 5\}) = 3$ in $M(Q)$. In other words, take any codeword $\mathbf{x} = (x_1, x_2, x_3, x_4, x_5) \in \mathbb{F}_2^5$ and view only the first or *the last three coordinates. In each case there is a vector in Q with the same values on the corresponding coordinates, so $d_{|T}(\mathbf{x}, Q) = 0\}$ for all $T \in \mathcal{T}$. In particular, there exist $\mathbf{x_1} = (x_1, x_2, x_3, x_1 + x_2, x_2 + x_3) \in Q$ and $\mathbf{x_2} = (x_4 - x_5 + x_3, x_5 - x_3, x_3, x_4, x_5) \in Q$.*

We are now ready to describe the star product scheme. Assume we want to download file x_i:

1. Construct the **query matrix** A^1 for the first iteration:

   1.1. Choose mb codewords from Q uniformly at random and assign them an arbitrary order denoting by $\mathbf{q}_{i_j}$. The codewords do not need to be unique.

   1.2. For each row j of file x_i, we add $\mathbf{w}_{ij} \notin Q$ such that $\mathbf{d}_{ij} + \mathbf{w}_{ij} \in Q^{\mathcal{T}}$.

   1.3. Set the rows of A^1 to be

   $$a_{lj} = \begin{cases} \mathbf{q}_{lj} & \text{if } l \neq i \\ \mathbf{q}_{lj} + \mathbf{w}_{ij} & \text{if } l = i \end{cases}.$$

2. By (4.1) the **response vector** R can be decomposed as

   $$R = \mathbf{v} + \sum_{i=1}^{b} y_{li} \star \mathbf{w}_{li},$$

   where $\mathbf{v}$ is a codeword in $C \star Q$, y_{li} is row l_i in the distributed storage system Y, and the vectors $y_{li} \star \mathbf{w}_{li}$ have pairwise disjoint supports in known positions.

3. Let H be a generator matrix of $(C \star Q)^{\perp}$. Then

   $$HR^{\text{tr}} = H\mathbf{v}^{\text{tr}} + H\left(\sum_{i=1}^{b} y_{li} \star \mathbf{w}_{li}\right)^{\text{tr}} = H\left(\sum_{i=1}^{b} y_{li} \star \mathbf{w}_{li}\right)^{\text{tr}}.$$

Note that in the star product schemes, downloading the entire collection of files means running the algorithm ms times, whereas downloading no files means $q_{i_j} \in Q$ for all i and j.

We will now illustrate the scheme with an example.

Example 4.5.2. *Let $Q \subseteq \mathbb{F}_3^6$ and $\mathcal{T} = \{\{1,2,3\},\{4,5,6\}$ and let*

$$G_C = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix},$$

and

$$Y = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 2 & 2 \end{bmatrix},$$

i.e. *the distributed storage system consists of four files and each file contains only one row. Assume we want to download x_2 and we choose $Q = \text{Rep}[6]$. Further, assume that, for all i, $\mathbf{q}_i = (1,1,1,1,1,1)$ was chosen uniformly at random from $\{(1,1,1,1,1,1),(0,0,0,0,0,0)\}$. Let $\mathbf{w}_2 = (0,0,0,1,1,1)$, so the query matrix becomes*

$$A = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Then the response vector is

$$R = (\langle a^1, y^1\rangle, \dots, \langle a^6, y^6\rangle) = (2,2,2,1,2,2) = (2,2,2,0,1,1) + (0,0,0,1,1,1).$$

Since Q is a repetition code, we know that $H = G_{(C\star Q)^\perp} = G_{C^\perp}$ and

$$G_{C^\perp} = \begin{bmatrix} -1 & -1 & -1 & 1 & 0 & 0 \\ -1 & -1 & 0 & 0 & 1 & 0 \\ 0 & -1 & -1 & 0 & 0 & 1 \end{bmatrix}.$$

The we get $HR^{\text{tr}} = H\mathbf{w}_2^{\text{tr}} = (1,1,1)^{\text{tr}}$, which correspond to (x_2^4, x_2^5, x_2^6). Furthermore, we know that for any x_i, $x_i^1 + x_i^2 + x_i^3 = x_i^4$, $x_i^1 + x_i^2 = x_i^5$, and $x_i^2 + x_i^3 = x_i^6$, so we can reconstruct the remaining symbols of x_2.

In this example, the storage code and the colluding pattern allowed us to download all information symbols during one iteration, so no further iterations are necessary.

We refer the reader to [10] for the proof of correctness and privacy of the star product scheme.

4.6 Efficiency of a PIR Scheme

By convention, the *rate* of a PIR scheme is used to evaluate its efficiency and is defined as the ratio of the size of file x_i to the total number of symbols that were downloaded, *i.e.* $R = \frac{bk}{ns}$. The reason why the formula does not include a reference to the queries is that, by assumption, the cost of uploading is negligible. We will be also using the notion of *dimension of the quotient* $Q^{\mathcal{T}}/Q$, which we will denote by $\rho(\mathcal{T}, Q)$.

Chapter 5

Lift over a collusion pattern

The choice of an efficient retrieval code Q depends on two factors:

1. the limitations set by the collusion pattern $\mathcal{T}$;
2. the algebraic properties of $C \star Q$.

The second factor is important because, even if $\rho(\mathcal{T}, Q) > 0$, it is possible that $\dim(C \star Q) = n$ and hence the download rate of the scheme is 0.

In this thesis, we will focus on the first factor. This can be also interpreted as C being a repetition $\mathrm{Rep}[n]$ code. Recall that $\mathrm{Rep}[n] \subset \mathbb{F}_q^n$ is a one-dimensional linear vector space generated by $\mathbf{1} = (1, \ldots, 1)$, so $\mathrm{Rep}[n] \star Q = Q$ and $\dim(\mathrm{Rep}[n] \star Q) = \dim Q$ for any linear code Q.

Before we tackle this directly, let us review the notion of $Q^{\mathcal{T}}$, the lift of the linear code Q over a collusion pattern $\mathcal{T}$. We introduced and defined $Q^{\mathcal{T}}$ in Section 4.5. Intuitively, it is the linear code that preserves the linear dependencies visible to the collusion pattern. However, it is still unclear exactly how $Q^{\mathcal{T}}$ is constructed.

5.1 Linear conditions observed by the collusion pattern

A good approach to describe the lift of a code is by listing the linear conditions that should be satisfied by all codewords in the lift. In other words,

$$Q^{\mathcal{T}} = \{\mathbf{x} \in \mathbb{F}_q^n : l(\mathbf{x}) = 0 \quad \forall l \in \mathcal{L}_{Q^{\mathcal{T}}}\},$$

where l is a linear condition and $\mathcal{L}_{Q^{\mathcal{T}}}$ is a generating set of $Q^{\mathcal{T}}$.

In particular, it is enough to limit the analysis to the minimal support linear conditions. This is because, for any linear code C,

$$l \in \mathcal{L}_{\text{all}} - \mathcal{L}_{\text{min}} \implies \exists\{l_1, \ldots, l_k\} \subseteq \mathcal{L}_{\text{min}} \text{ s.t. } l = \sum_{i=1}^{k} a_i l_i,$$

where $\mathcal{L}_{\text{all}}$ is the set of all linear conditions satisfied by C, $\mathcal{L}_{\text{min}} \subseteq \mathcal{L}_{\text{all}}$ is the set of minimal support linear conditions and for some $a_i \in \mathbb{F}_q - \{0\}$. In other words, the minimal support codewords in $C^{\perp}$ generate $C^{\perp}$.

Going forward, we will assume that $\mathcal{L}_{Q^{\mathcal{T}}}$ is a basis, *i.e.*, it only contains minimal support conditions that are sufficient to generate $Q^{\mathcal{T}}$ and has minimal cardinality. Combinatorially, we want to describe the matroid $M(Q^{\mathcal{T}})$ by its circuits.

There are two equivalent ways to identify a linear condition on a given support S of cardinality s. We will describe them here and use later when we introduce our algorithm for constructing $Q^{\mathcal{T}}$ in Section 5.2.

The first way is by checking whether there exists a dual codeword $\mathbf{y} \in Q^{\perp}$ such that $\text{supp}(\mathbf{y}) = S$. Note that if no such codeword exists, then $\mathbf{x} \in Q^{\mathcal{T}} - Q$ can take arbitrary values on the coordinates in S, *i.e.*, $Q^S = \mathbb{F}_q^s$. We show when such a codeword exists in the following proposition.

Proposition 1. *Let $C \subsetneq \mathbb{F}_q^n$ be a linear code and S a subset of its coordinates. Then there exists a linear condition on S if and only if there exists a dual codeword $\mathbf{y}$ with $\text{supp}(\mathbf{y}) = S$.*

Proof. Recall that

$$C^{\perp} = \{\mathbf{y} \in \mathbb{F}_q^n \mid \forall \mathbf{x} \in C \ \ \langle \mathbf{x}, \mathbf{y} \rangle = 0\}.$$

But this is equivalent to saying that there exists a linear condition on $\text{supp}(\mathbf{y})$ that is satisfied by all $\mathbf{x} \in C$. Then this is also a linear condition in C. □

The second way is by fixing some generator matrix G_Q for Q and taking its row-reduced restriction on S, which we denote by $(G_Q)_{|S}$. If $(G_Q)_{|S} \equiv I_s$, then $Q^S = \mathbb{F}_q^s$. On the other hand, if $(G_Q)_{|S} \not\equiv I_s$, then there exist linear conditions supported on S. In particular, if $(G_Q)_{|S} \equiv (I_{s-1}|l)$, where l is a column with non-zero entries, then S corresponds to the support of a minimal linear condition, and the exact coefficients of the corresponding minimal linear condition are given by l. We show why this is true in the following proposition.

Proposition 2. *Let C be a linear code with a generator matrix G_C. Assume that for some $S \subseteq E$ with $|S| = s$, it holds that $(G_C)_{|S} \equiv (I_{s-1}|l)$ with all $l_i \neq 0$. We denote the index of the coordinates of the elements of S by s_i, where s_i is the index of the ith element of S in the ground set E. Then there exists a minimal support linear condition on S given by*

$$l_1 x_{s_1} + \cdots + l_{s-1} x_{s_{s-1}} = x_{s_s}.$$

Proof. We puncture C on $E - S$ and denote the new code by C_{punct}. Then S is a circuit in $M(C_{\text{punct}})$. But $M(C_{\text{punct}}) = M(C) \backslash (E - S)$, so it is also a circuit in $M(C)$. Therefore, there exists a minimal support linear condition on S. The coefficients of this linear condition follow immediately from the row-reduced restriction $(G_C)_{|S} \equiv (I_{s-1}|l)$ with all $l_i \neq 0$. □

Let us illustrate these approaches with an example.

Example 5.1.1. *Let C be a $[6,3]$ MDS code over $\mathbb{F}_7$ and $G_{C^\perp}$ be a generator matrix of its dual $C^\perp$*

$$G_{C^\perp} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 & 6 \end{bmatrix}.$$

Then its row-reduced form, $R_{C^\perp}$, is

$$R_{C^\perp} = \begin{bmatrix} 1 & 0 & 0 & 6 & 3 & 2 \\ 0 & 1 & 0 & 1 & 3 & 4 \\ 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix},$$

so C has a generator matrix

$$G_C = \begin{bmatrix} 1 & 6 & 6 & 1 & 0 & 0 \\ 4 & 4 & 6 & 0 & 1 & 0 \\ 5 & 3 & 6 & 0 & 0 & 1 \end{bmatrix}.$$

Let the set S be $\{1,2,5,6\}$. First, let us check the linear conditions on S by identifying a dual codeword $\mathbf{y}$ such that $\operatorname{supp}(\mathbf{y}) = S$. It is easy to see that

$$\mathbf{y} = (1,1,0,0,6,6) = (1,1,1,1,0,0) - (0,0,1,1,1,1),$$

i.e., $x_1 + x_2 + 6x_5 + 6x_6 = 0$.

Now let us identify the same linear condition using the restriction $(G_C)_{|S}$.

$$(G_C)_{|S} = \begin{bmatrix} 1 & 6 & 0 & 0 \\ 4 & 4 & 1 & 0 \\ 5 & 3 & 0 & 1 \end{bmatrix} \cong \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 6 \end{bmatrix},$$

and we also get $x_1 + x_2 + 6x_5 + 6x_6 = 0$.

We still need to show that the two approaches are equivalent.

Lemma 4. *Let V be a vector space over $\mathbb{F}$ and $\mathbf{x} \in V$ a minimal support vector. Then for every $\mathbf{y} \in V$ such that $\operatorname{supp}(\mathbf{y}) = \operatorname{supp}(\mathbf{x})$, there exists an $\alpha \in \mathbb{F}$ such that $\mathbf{y} = \alpha\mathbf{x}$.*

Proof. For contradiction, assume no such α exists. Let $y_1 = \beta x_1$ where $\beta \in \mathbb{F}$. Then $\mathbf{0} \neq \mathbf{y} - \beta\mathbf{x} \in V$ and

$$\operatorname{supp}(\mathbf{y} - \beta\mathbf{x}) \subsetneq \operatorname{supp}(\mathbf{y}).$$

This contradicts the minimality of the support of $\mathbf{y}$. □

Proposition 3. *Let C be a linear code with a generator matrix G_C. Then the following are equivalent:*

1. *For some $S \subseteq E$ with $|S| = s$, it holds that $(G_C)_{|S} \equiv (I_{s-1}|l)$ with all $l_i \neq 0$;*
2. *There exists a minimal support dual codeword $\mathbf{y}$ such that $\operatorname{supp}(\mathbf{y}) = S$ and $\mathbf{y} = \alpha(l_1, \ldots, l_{s_1}, -1)$, $\alpha \in \mathbb{F}_q \backslash \mathbf{0}$.*

Proof. This is a corollary of Propositions 1 and 2. The first statement is equivalent to saying that S is a circuit in $M(C)$ but this is also equivalent to having a minimal support dual codeword on S. Since C is a vector space, the minimal support dual codewords are unique up to scalar multiplication by Lemma 4. □

Clearly, if $l \in \mathcal{L}_{Q^{\mathcal{T}}}$, then $l \in \mathcal{L}_Q$. Therefore, we only need to identify those conditions that will be inherited from Q. An easy place to start is by looking at those that are supported on the collusion pattern $\mathcal{T}$. This is because, if, for some $T \in \mathcal{T}$, $\sum_{i \in T} a_i x_i = 0$ is satisfied by all $\mathbf{x} \in Q$, then it is also satisfied by all $\mathbf{x} \in Q^{\mathcal{T}}$. We say that these linear conditions are *observed* by $\mathcal{T}$.

Let $\mathcal{L}$ be the collection of minimal linear conditions observed by $\mathcal{T}$. We need to answer three questions:

1. How to construct a linear code L determined by $\mathcal{L}$?
2. Is $L = Q^{\mathcal{T}}$?
3. To what extent does $M(Q^{\mathcal{T}})$ depend on the algebraic properties of Q?

We are interested in the last question because $\mathcal{T}$ is a combinatorial construction (abstract simplicial complex) and one would hope that the construction of $Q^{\mathcal{T}}$ is also a combinatorial operation. Therefore, we will address the first two questions in this section and cover the last question in Section 6.2.

5.2 Construction of a linear code L from the observed minimal linear equations $\mathcal{L}$

We are now ready to construct a generator matrix for the linear code L from the collection of minimal linear conditions $\mathcal{L}$:

1. Identify the set $\mathcal{L}$ of observed minimal support linear conditions using restrictions of G_Q on each colluding set;
2. Form a matrix A_L whose rows are dual codewords corresponding to the linear conditions in $\mathcal{L}$;
3. Row reduce and bring A_L to the systematic form without changing the label of the columns (although their order might change). This is a generator matrix $G_{L^\perp} = (I_{n-k} \quad P)$ for the dual code $L^\perp$;
4. Produce a generator matrix $G_L = (-P^{\text{tr}} \quad I_k)$ for L
5. Permute the columns of G_L so their order corresponds to their index in Step 3.

Alternatively, in the last step, one could relabel the elements of colluding sets to reflect their new position in the distributed storage system.

Example 5.2.1. *Let Q be the $RS_7[7,3]$ code whose canonical generator matrix is*

$$G_Q = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ 0 & 1 & 4 & 2 & 2 & 4 & 1 \end{bmatrix}.$$

For future reference, we denote the reduced row echelon form of G_Q by R_Q

and it is

$$R_Q = \begin{bmatrix} 1 & 0 & 0 & 1 & 3 & 6 & 3 \\ 0 & 1 & 0 & 4 & 6 & 6 & 4 \\ 0 & 0 & 1 & 3 & 6 & 3 & 1 \end{bmatrix}.$$

Let the maximal elements of $\mathcal{T}$ *be* $\{\{1,2,3,4\},\{2,3,5,6\},\{4,5,6,7\}\}$*. Then the collection of observed minimal support linear equations is*

$$\mathcal{L} = \{x_1 + 4x_2 + 3x_3 = x_4, \quad x_2 + 5x_3 + 2x_5 = x_6, \quad x_4 + 4x_5 + 3x_6 = x_7\},$$

which corresponds to matrix

$$A_L = \begin{array}{c} \begin{array}{ccccccc} i & ii & iii & iv & v & vi & vii \end{array} \\ \begin{bmatrix} 1 & 4 & 3 & 6 & 0 & 0 & 0 \\ 0 & 1 & 5 & 0 & 2 & 6 & 0 \\ 0 & 0 & 0 & 1 & 4 & 3 & 6 \end{bmatrix} \end{array}.$$

After row reduction we get

$$G_{L^\perp} = \begin{array}{c} \begin{array}{ccccccc} i & ii & iv & iii & v & vi & vii \end{array} \\ \begin{bmatrix} 1 & 0 & 0 & 4 & 3 & 0 & 6 \\ 0 & 1 & 0 & 5 & 2 & 6 & 0 \\ 0 & 0 & 1 & 0 & 4 & 3 & 6 \end{bmatrix} \end{array}.$$

This gives a generator matrix for the linear code L:

$$G_L = \begin{array}{c} \begin{array}{ccccccc} i & ii & iv & iii & v & vi & vii \end{array} \\ \begin{bmatrix} 3 & 2 & 0 & 1 & 0 & 0 & 0 \\ 4 & 5 & 3 & 0 & 1 & 0 & 0 \\ 0 & 1 & 4 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array}.$$

Resorting the columns gives

$$G_L = \begin{array}{c} \begin{array}{ccccccc} i & ii & iii & iv & v & vi & vii \end{array} \\ \begin{bmatrix} 3 & 2 & 1 & 0 & 0 & 0 & 0 \\ 4 & 5 & 0 & 3 & 1 & 0 & 0 \\ 0 & 1 & 0 & 4 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \end{array}.$$

We denote the row-reduced form by R_L and it is

$$R_L = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 4 & 0 & 1 & 0 \\ 0 & 0 & 1 & 3 & 0 & 5 & 4 \\ 0 & 0 & 0 & 0 & 1 & 2 & 3 \end{bmatrix}.$$

To improve our understanding of the structure of G_L, let us compare the reduced row echelon forms of G_L and G_Q, denoted by R_L and R_Q. This time we will start with our example and then generalize the statement. Recall we use subscripts to denote rows and superscripts to denote columns.

Example 5.2.2. *We have*

$$R_L = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 4 & 0 & 1 & 0 \\ 0 & 0 & 1 & 3 & 0 & 5 & 4 \\ 0 & 0 & 0 & 0 & 1 & 2 & 3 \end{bmatrix},$$

$$R_Q = \begin{bmatrix} 1 & 0 & 0 & 1 & 3 & 6 & 3 \\ 0 & 1 & 0 & 4 & 6 & 6 & 4 \\ 0 & 0 & 1 & 3 & 6 & 3 & 1 \end{bmatrix}.$$

We will now multiply $(R_L)_4$ by $(R_Q)_i^5$ and then replace $(R_L)_i$ by $(R_L)_i + (R_Q)_i^5(R_L)_4$, we get

$$G_L = \begin{bmatrix} 1 & 0 & 0 & 1 & 3 & 6 & 3 \\ 0 & 1 & 0 & 4 & 6 & 6 & 4 \\ 0 & 0 & 1 & 3 & 6 & 3 & 1 \\ 0 & 0 & 0 & 0 & 1 & 2 & 3 \end{bmatrix}.$$

In general, consider the generator matrix G_Q of Q in the standard form:

$$G_Q = \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & p_{1,k+1} & \cdots & p_{1,n} \\ 0 & \ddots & 0 & \vdots & \ddots & \vdots \\ 0 & 0 & 1 & p_{k,k+1} & \cdots & p_{k,n} \end{array}\right].$$

Then after a permutation of columns, the generator matrix G_L of L is of the form

$$\begin{aligned} G_L &= \begin{bmatrix} 1 & 0 & 0 & p_1^{k+1} & \cdots & p_1^{k+m} & p_1^{k+m+1} & \cdots & p_1^n \\ 0 & \ddots & 0 & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 1 & p_k^{k+1} & \cdots & p_k^{k+m} & p_k^{k+m+1} & \cdots & p_k^n \\ 0 & 0 & 0 & 1 & \cdots & 0 & (p_L)_{k+1}^{k+m+1} & \cdots & (p_L)_{k+1}^n \\ 0 & \ddots & 0 & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 & (p_L)_{k+m}^{k+m+1} & \cdots & (p_L)_{k+m}^n \end{bmatrix} \\ &= \begin{bmatrix} G_Q \\ G_{L/Q} \end{bmatrix} \end{aligned}$$

and $\dim L = k + m$. This block structure is not surprising, since $Q \subseteq L$ by construction.

Then R_L, row-echelon form of G_L, is

$$R_L = \left[\begin{array}{cc|c} I_k & 0 & P' \\ 0 & I_m & P_L \end{array} \right],$$

where $(P')_i^j = P_i^j - \sum_{l=k+1}^{k+m} P_i^l (P_L)_l^j$ and P_L is given by the algorithm above.

5.3 Equivalence of L and Q^T

The following proposition shows the uniqueness of G_L and the equivalence of L and Q^T.

Proposition 4. *G_L is unique up to elementary row operations and $L = Q^{\mathcal{T}}$.*

Proof. G_L is constructed using standard linear algebra to be a generator matrix of the vector space $L = \{\mathbf{x} \in \mathbb{F}_q^n \text{ for all } l \in \mathcal{L}\}$, so it is unique up to elementary row operations.

Next, we want to show that L is indeed the lift of Q over $\mathcal{T}$. Since we included all linear equations corresponding to $\mathcal{C} \cap \mathcal{T}$, we know that if $\mathbf{x} \in L$, then $d_{|T}(\mathbf{x}, L) = 0$ for all $T \in \mathcal{T}$, so $\mathbf{x} \in Q^{\mathcal{T}}$ and hence $L \subseteq Q^{\mathcal{T}}$. On the other hand, if $L \subsetneq Q^{\mathcal{T}}$, then it means that at least one of the linear conditions introduced in L is redundant. Let S be the support of that linear condition. Since the linear condition was redundant, $Q^{\mathcal{T}}_{|S} = \mathbb{F}_q^{|S|}$. However, by construction, $S \in \mathcal{T}$ and so $Q^{\mathcal{T}}_{|S} \subsetneq \mathbb{F}_q^{|S|}$. Therefore, L contains only the necessary linear conditions and $L = Q^{\mathcal{T}}$. $\square$

Chapter 6

Combinatorics of $Q^{\mathcal{T}}$

Now that we have shown how to construct $Q^{\mathcal{T}}$, we want to analyze its combinatorics. Our goal is to show that the operation of lifting a code over a collusion pattern is not matroid invariant and to discuss the properties that determine the matroid structure of $Q^{\mathcal{T}}$ completely or partially.

6.1 Elementary properties

Let us start by reviewing some of the basic combinatorial properties of the lift $Q^{\mathcal{T}}$ that are induced by an arbitrary collusion pattern $\mathcal{T}$. It follows immediately from the definition that $\mathcal{I} \subseteq \mathcal{I}^{\mathcal{T}}$ and $Q^{T} = \mathbb{F}_q^{|T|}$ whenever $T \in \mathcal{I}$.

Since we know that $Q \subseteq Q^{\mathcal{T}} \subseteq \mathbb{F}_q^n$, we would like to know when the statements hold with equality. It is easier to start with the second one. In what follows, $\mathcal{C}$ denotes the set of circuits of $M(Q)$.

Lemma 5. *The following are equivalent:*

1. $Q^{\mathcal{T}} = \mathbb{F}_q^n$*;*
2. $\mathcal{C} \cap \mathcal{T} = \emptyset$*;*

Proof. Every $T \in \mathcal{T}$ is independent in Q if and only if $d_{|T}(\mathbf{x}, Q) = 0$ for all $T \in \mathcal{T}$ and $\mathbf{x} \in \mathbb{F}_q^n$. This holds if and only if $Q^{\mathcal{T}} = \mathbb{F}_q^n$. □

Corollary 2. *If Q is an $[n, k]$ MDS code and for all T, $|T| \leq k$, then $Q^{\mathcal{T}} = \mathbb{F}_q^n$.*

When it comes to the first containment, one could hope that $\mathcal{C}^{\mathcal{T}} = \mathcal{C} \cap \mathcal{T}$, as it is in the following example:

Example 6.1.1. *Let*

$$G_Q = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}$$

and $\mathcal{T} = \{\{1,2,3\},\{4,5,6\}\}$*. The set* $\{3,4\}$ *is a circuit but it is independent in* $M(Q^{\mathcal{T}})$ *for there exists a codeword* $\mathbf{d}^{\mathcal{T}} \in Q^{\mathcal{T}}$ *corresponding to an arbitrary pair of values on* $\{3,4\}$*.*

However, $\mathcal{C}^{\mathcal{T}} = \mathcal{C} \cap \mathcal{T}$ is not true in general. Consider the following setting:

Example 6.1.2. *Let* $Q = \mathrm{Rep}[5]$*, so every pair of coordinates is a circuit. Now, assume all pairs except* $\{1,2\}$ *collude. Since* $\{1,5\}$ *collude, we know that* $d_1^{\mathcal{T}} = d_5^{\mathcal{T}}$ *for all* $\mathbf{d}^{\mathcal{T}} \in Q^{\mathcal{T}}$*. Similarly,* $d_2^{\mathcal{T}} = d_5^{\mathcal{T}}$*. Hence,* $d_1^{\mathcal{T}} = d_2^{\mathcal{T}}$ *and* $d_{|\{1,2\}}(Q, Q^{\mathcal{T}}) = 0$*.*

Therefore, we will only give partials result related to this containment.

Lemma 6. *If* $\mathcal{C} \subseteq \mathcal{T}$*, then* $Q = Q^{\mathcal{T}}$*.*

Proof. If C is a circuit in Q and a colluding set, then $d_{|C}(Q, Q^{\mathcal{T}}) = 0$, so $C \in \mathcal{C}^{\mathcal{T}}$. Hence, if $\mathcal{C} \subseteq \mathcal{T}$, then $M(Q) = M(Q^{\mathcal{T}})$ and the statement follows. □

Lemma 7. *The following are equivalent:*

- $Q^{\mathcal{T}} = Q$*;*
- *Either* $Q = \mathbb{F}_q^n$ *or every linear dependency in* Q *is a linear combination of the linear conditions defined on* $\mathcal{C} \cap \mathcal{T}$*.*

Proof. We have two cases: either $Q = \mathbb{F}_q^n$ or $Q \subsetneq \mathbb{F}_q^n$. In the first case, the statements follow immediately from $Q \subseteq Q^{\mathcal{T}} \subseteq \mathbb{F}_q^n$.

If $Q \subsetneq \mathbb{F}_q^n$, then the statement is equivalent to saying that $Q = L$, where L is the linear code from Chapter 5 that is generated by the linear conditions supported on $\mathcal{C} \cap \mathcal{T}$. The claim then follows from Proposition 4. □

6.2 Matroid of $Q^{\mathcal{T}}$

The next step in understanding the combinatorics of lifting is to analyze the relationship between the matroids of the code Q and its lift $Q^{\mathcal{T}}$. In general, one can define multiple linear codes Q with the same matroid structure. For example, all $[n,k]$ MDS codes are representations of the uniform matroid $U_{k,n}$.

At this point, it is good to mention that we can assume that for every $i \in [n]$, there exists a colluding set T that properly contains i. This is because if for some i no such colluding set exists, then there exists a Q such that $\rho(\mathcal{T}, Q) \geq 1$. To see that we can choose $j \neq i$ and for all codewords $\mathbf{d} \in Q$, let $d_j = d_i$. Then $Q^{\mathcal{T}} = (Q_{|[n]-\{i\}})^{\mathcal{T}} \times \mathbb{F}_q$, so, in order to understand all $\mathcal{T}$, it is sufficient to understand only those that do not contain any non-colluding servers.

6.2.1 Matroid invariance of $Q^{\mathcal{T}}$

Since a collusion pattern $\mathcal{T}$ is an abstract simplicial complex and the operation of lifting has a strong combinatorial flavor, it may appear that lifts are matroid invariant. This would mean that for any two different linear codes Q_1 and Q_2, $M(Q_1) = M(Q_2)$ implied $M(Q_1^{\mathcal{T}}) = M(Q_2^{\mathcal{T}})$. However, this is not true as the following example illustrates.

Example 6.2.1. *Let Q_1 and Q_2 be two $[6,3]$ MDS codes over $\mathbb{F}_7$ as follows. A generator matrix for the dual code $Q_1^{\perp}$ is*

$$G_{Q_1^{\perp}} = \begin{bmatrix} 1 & 2 & 1 & 5 & 0 & 0 \\ 1 & 5 & 0 & 0 & 5 & 1 \\ 0 & 0 & 5 & 1 & 2 & 1 \end{bmatrix}$$

and a generator matrix for the dual code $Q_2^{\perp}$ is

$$G_{Q_2^{\perp}} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 & 6 \end{bmatrix}.$$

Clearly, $M(Q_1) = M(Q_2)$. Assume the collusion pattern is

$$\mathcal{T} = \{1234, 1256, 3456\}.$$

In Q_1, the unique dual codewords supported on the facets of $\mathcal{T}$ are

$$\{(1,2,1,5,0,0),(1,5,0,0,5,1),(0,0,5,1,2,1)\}$$

and they form a basis of $Q_1^\perp$, so $Q_1 = Q_1^{\mathcal{T}}$. Observe that uniqueness follows from the fact that the facets of $\mathcal{T}$ are circuits in $M(Q_1)$.

On the other hand, in Q_2, the corresponding codewords are

$$\{(1,1,1,1,0,0),(0,0,1,1,1,1),(1,1,0,0,6,6)\},$$

where

$$(1,1,0,0,6,6) = (1,1,1,1,0,0) - (0,0,1,1,1,1),$$

so $Q_1 \subsetneq Q_1^{\mathcal{T}}$ and uniqueness again follows from the fact that the facets are circuits in $M(Q_2)$.

At this stage, we will state a partial result relating the matroid of the lift to the matroid of the retrieval code.

Proposition 5. *Let Q_1 and Q_2 be two different linear codes such that $M(Q_1) = M(Q_2)$ and for every basis of $Q_1^\perp$, there is a corresponding basis on the same set of supports in $Q_2^\perp$. Then for any collusion pattern $\mathcal{T}$, $M(Q_1^{\mathcal{T}}) = M(Q_2^{\mathcal{T}})$.*

Proof. Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be two collections of minimal support linear conditions on Q_1 and Q_2 that are generating sets of $Q_1^\perp$ and $Q_2^\perp$, respectively, and such that their elements have the same supports. By assumption, such sets exist. We assign an arbitrary order to the set of supports and use that order to index the linear conditions in $\mathcal{L}_1$ and $\mathcal{L}_2$.

Let $\sum_{i=1}^n \alpha_i^j x_i$ be the jth element of $\mathcal{L}_1$ and $\sum_{i=1}^n \beta_i^k x_i$ be the kth element of $\mathcal{L}_2$. We say that the linear map $\lambda_1^j = (\alpha_1^j, \dots, \alpha_n^j)$ is the corresponding jth linear map and $\lambda_2^k = (\beta_1^k, \dots, \beta_n^k)$ is the corresponding kth linear map of $\mathcal{L}_1$ and $\mathcal{L}_2$, respectively. We denote the collections of corresponding linear maps by Λ_1 and Λ_2 and define vector spaces $V_1 = \langle \lambda_1^j \mid \lambda_1^j \in \Lambda_1 \rangle$ and $V_2 = \langle \lambda_2^j \mid \lambda_2^j \in \Lambda_2 \rangle$. Observe that the sets of supports of vectors in V_1 and V_2 are equal to the sets of dependent sets of $M(Q_1)$ and $M(Q_2)$, respectively.

Now, let $\mathcal{L}_1^{\mathcal{T}} \subseteq \mathcal{L}_1$ and $\mathcal{L}_2^{\mathcal{T}} \subseteq \mathcal{L}_2$ be the subsets that are observed by $\mathcal{T}$ and $V_1^{\mathcal{T}} \subseteq V_1$, $V_2^{\mathcal{T}} \subseteq V_2$ be the vector spaces generated by $\Lambda_1^{\mathcal{T}}$ and $\Lambda_2^{\mathcal{T}}$.

Assume $M(Q_1^{\mathcal{T}}) \neq M(Q_2^{\mathcal{T}})$, so there exists a set S that is dependent in $M(Q_1^{\mathcal{T}})$ but independent in $M(Q_2^{\mathcal{T}})$. If S is dependent in $M(Q_1^{\mathcal{T}})$, then there exists a linear map λ_1^s with support on S that is an element of $V_1^{\mathcal{T}}$. Then for some $\lambda_1^k \in \Lambda_1^{\mathcal{T}}$, $\Lambda_1^{\mathcal{T}} \backslash \lambda_1^k \cup \lambda_1^s$ is a basis of $V_1^{\mathcal{T}}$. Furthermore, this basis can be extended to a basis of V_1. But then there exists a basis of V_1 defined on the same set of supports, which is a contradiction. □

6.2.2 Explicit construction of $\mathcal{C}^{\mathcal{T}}$

In an earlier version of the thesis, we attempted to construct $\mathcal{C}^{\mathcal{T}}$ directly from $\mathcal{C} \cap \mathcal{T}$. To accomplish this we considered the following question: given two circuits C' and C'' in $C^{\mathcal{T}}$, is an arbitrary dependent set S in Q also a dependent set in $Q^{\mathcal{T}}$? Note that if $S - (C' \cup C'') \neq \emptyset$, then one cannot conclude whether S is dependent in $Q^{\mathcal{T}}$ based on C', C'' alone because it is possible to extend any $\mathbf{x} \in \mathbb{F}_q^{S-(C' \cup C'')}$ to a codeword in $Q^{\mathcal{T}}$ by concatenating $\mathbf{x}$ with any $\mathbf{y} \in Q_{|C' \cup C''}$.

It is clear that $\mathcal{C}^{\mathcal{T}} = (\mathcal{C} \cap \mathcal{T}) \cup \mathcal{P} \cup \mathcal{N}$, where $\mathcal{P} := (\mathcal{C} \cap \mathcal{C}^{\mathcal{T}}) - (\mathcal{C} \cap \mathcal{T})$ and $\mathcal{N} := \mathcal{C}^{\mathcal{T}} - \mathcal{C}$. We say that the elements of $\mathcal{P}$ are the circuits of $M(Q)$ that are *preserved* in $M(Q^{\mathcal{T}})$ and the elements of $\mathcal{N}$ are the *new* circuits introduced in $M(Q^{\mathcal{T}})$.

We introduce the following notation. Let $C_i, C_j \in \mathcal{C}^{\mathcal{T}}$, and $C_i \cap C_j \neq \emptyset$, we say that a circuit $C \in \mathcal{C}^{\mathcal{T}}$ is *generated* by C_i, C_j, if $C \subseteq C_i \cup C_j$ and $C \in \mathcal{C}^{\mathcal{T}'}$, where $\mathcal{T}' = \{C_i, C_j\}$. We denote the set of all circuits generated by C_i, C_j as $\langle C_i, C_j \rangle$ and we know that $\langle C_i, C_j \rangle \neq \emptyset$ by the third circuit axiom.

Let $\mathcal{C} \cap \mathcal{T} = \{C_1, \ldots, C_k\}$. Then define

$$\langle C_1, \ldots, C_k \rangle := \bigcup_{i,j \in [k]} \langle C_i, C_j \rangle = \{C_1, \ldots, C_{k_1}\} \supseteq \{C_1, \ldots, C_k\}$$

and

$$\langle C_1, \ldots, C_k \rangle^2 := \bigcup_{i,j \in [k_1]} \langle C_i, C_j \rangle.$$

In general,

$$\langle C_1, \ldots, C_k \rangle^n := \bigcup_{i,j \in [k_{n-1}]} \langle C_i, C_j \rangle$$

and we set

$$\langle C_1, \ldots, C_k \rangle^0 := \{C_1, \ldots, C_k\}.$$

Assuming that these operations are well-defined, the hope is that

$$\mathcal{C}^{\mathcal{T}} = \lim_{n \to \infty} \langle \mathcal{C} \cap \mathcal{T} \rangle^n.$$

At the first glance, this approach to generation appears to be excessively complicated. However, it turns out to be relatively nice in the case of preserved circuits.

6.2.2.1 Preserved circuits

Let $C \in \mathcal{C}$, $C', C'' \in \mathcal{C}^{\mathcal{T}}, C' \cap C'' \neq \emptyset$ and $C \subseteq C' \cup C''$. When is $C \in \mathcal{C}^{\mathcal{T}}$? We need to consider three cases (see Table 6.1):

1. $C \subsetneq C' \triangle C''$;
2. $C \supseteq C' \triangle C''$;
3. $C \subsetneq C' \triangle C''$, $C \supsetneq C' \triangle C''$.

Case 1

Let $C \subsetneq C' \triangle C''$ and assume C is independent in $M(Q^{\mathcal{T}})$. By construction, there exists $e \notin C' \triangle C''$ and, without loss of generality, we can assume that $e \in C'$. We pick arbitrary values for elements in $C'' - C'$ and choose the elements of $C' \cap C''$ such that the linear condition of C'' is satisfied. We know that $C' - e$ is an independent set, so we can pick arbitrary values for $C - C''$ and since $e \notin C' \triangle C''$, we can set e to satisfy the linear condition of C'. Since none of the linear conditions of the known circuits is violated, C is indeed an independent set in $M(Q^{\mathcal{T}})$.

Case 2

Let $C \supseteq C' \triangle C''$ and assume $C \in \mathcal{I}^{\mathcal{T}}$. We pick arbitrary values for elements in $C'' - C'$ and choose the elements of $C' \cap C''$ such that the linear condition of C'' is satisfied. However, $C - C'' = C' - C''$, so setting arbitrary values to $C - C''$ will violate the linear condition of C'. Hence, C is a dependent subset in $M(Q^{\mathcal{T}})$. Furthermore, every proper subset of C is independent in $M(Q)$ and by the first property above, this means that every proper subset of C is independent in $M(Q^{\mathcal{T}})$, so $C \in \mathcal{C}^{\mathcal{T}}$.

Case 3

Let $C \subsetneq C' \triangle C''$, $C \supsetneq C' \triangle C''$ and assume $C \in \mathcal{I}^{\mathcal{T}}$. This case is essentially equivalent to Case 1, because there exists $e \in C' \triangle C'' - C$ and by performing the same analysis, we can see that both linear conditions of C' and C'' will be satisfied even if the elements of C have arbitrary values. Therefore, C is indeed an independent set in $M(Q^{\mathcal{T}})$.

We summarise the findings in the following proposition.

Case	Observed circuits	Unobserved circuits
1: $C \subsetneq C'\triangle C''$	$x_1 + x_2 + x_3 = x_4$, $x_3 + x_5 + x_6 = x_4$	$x_1 = x_5$, $x_2 = x_6$
2: $C \supseteq C'\triangle C''$	$x_1 + x_2 + x_3 = x_4$, $x_3 + x_5 = x_4$	$x_1 + x_2 = x_5$
	$x_1 + x_2 + x_3 = x_4$, $x_3 + x_4 = x_5$	$x_1 + x_2 + 2x_3 = x_5$
3: $C \not\subseteq C'\triangle C''$, $C \not\supseteq C'\triangle C''$	$x_1 + x_2 + x_3 + x_4 = x_5$, $x_3 + x_4 + x_5 + x_6 + x_7 = x_8$	$x_2 + x_4 = x_6$, $2x_1 + x_2 + x_3 + x_4 = x_7$

Table 6.1: Examples of the three cases discussed in Section 6.2.2.1. All linear equations are defined over $\mathbb{Q}$.

Proposition 6. *Let $C \in \mathcal{C}$, $C', C'' \in \mathcal{C}^{\mathcal{T}}, C' \cap C'' \neq \emptyset$ and $C \subseteq C' \cup C''$. Then*

1. *$C \subsetneq C'\triangle C'' \implies C \in \mathcal{I}^{\mathcal{T}}$;*
2. *$C \supseteq C'\triangle C'' \implies C \in \mathcal{C}^{\mathcal{T}}$;*
3. *$C \not\subseteq C'\triangle C''$, $C \not\supseteq C'\triangle C'' \implies C \in \mathcal{I}^{\mathcal{T}}$.*

Since we are working with representable matroids, the circuits are given by minimal support linear equations. In general, two linear equations with intersecting supports do not need to have the same coefficients on the elements of the intersection. Hence, one may wonder how having different coefficients affects the matroid.

Proposition 7. *Suppose two intersecting circuits C', C'' of $M(Q^{\mathcal{T}})$ are given by linear equations*

$$\sum_{i \in C'-C''} \alpha_i x_i + \sum_{i \in C' \cap C'' - \{x_j\}} \alpha_i x_i = x_j$$

and

$$\sum_{i \in C''-C'} \beta_i x_i + \sum_{i \in C' \cap C'' - \{x_j\}} \beta_i x_i = x_j,$$

where x_j is an arbitrary element in the intersection. Let $S \subseteq C' \cap C'' - \{x_j\}$ be the set where $\alpha_i \neq \beta_i$. Then $S \cup (C'\triangle C'')$ is a dependent set in $M(Q^{\mathcal{T}})$.

Proof. We have

$$\sum_{i \in C'-C''} \alpha_i x_i + \sum_{i \in S} (\alpha_i - \beta_i) x_i = \sum_{i \in C''-C'} \beta_i x_i,$$

so $S \cup (C' \triangle C'')$ is a dependent set in $M(Q^{\mathcal{T}})$. □

Corollary 3. *If $C' \triangle C''$ is a circuit in $M(Q)$, then $\alpha_i = \beta_i$ for all $i \in C' \cap C'' - \{x_j\}$.*

In other words, our analysis suggests that we need to focus on analyzing the symmetric differences of known circuits of $\mathcal{C}^{\mathcal{T}}$. In the case of preserved circuits it really is enough since, once we know that a circuit C of $M(Q)$ is a dependent set in $M(Q^{\mathcal{T}})$, we know that it is also a circuit in $M(Q^{\mathcal{T}})$. This is because every proper subset of C is independent in $M(Q)$ and, hence, also in $M(Q^{\mathcal{T}})$.

6.2.2.2 New circuits

Now consider the setting when $C', C'' \in \mathcal{C}^{\mathcal{T}}$, and $S \subseteq C' \cup C''$ is a dependent set in $M(Q)$ but not a circuit. Clearly, if at least one of the circuits contained in S are preserved in $M(Q^{\mathcal{T}})$, then S is also a dependent set in $M(Q^{\mathcal{T}})$. So assume no such circuit exists and we are asking whether this implies that S is a circuit in $M(Q^{\mathcal{T}})$.

The previous analysis suggests that we need to analyze the relationship between S and $C' \triangle C''$. Unfortunately, this approach does not extend well to this setting, because the notion $\langle C_i, C_j \rangle$ of "generated circuits" depends not only on the sets C_i, C_j but also on the entire collection $\mathcal{C}$. We will illustrate this with an example.

Example 6.2.2. *Assume that after some operations we conclude that the following sets are circuits in $M(Q^{\mathcal{T}})$:*

$$C_1 = \{x_1, x_2, x_3\}, \quad C_2 = \{x_3, x_4, x_5\}, \quad C_3 = \{x_4, x_5, x_6, x_7\},$$
$$C_4 = \{x_1, x_2, x_7, x_8\}.$$

They could, for example, correspond to linear conditions

$$\{x_1 + x_2 = x_3, x_3 + x_4 = x_5, x_5 - x_4 + x_6 = x_7, x_1 + x_2 + x_8 = x_7\}.$$

Then

$$C_1 \triangle C_2 = \{x_1, x_2, x_4, x_5\}$$

and

$$C_3 \triangle C_4 = \{x_1, x_2, x_4, x_5, x_6, x_8\},$$

so $(C_1 \triangle C_2) \subsetneq (C_3 \triangle C_4)$. However, this fact would not be clear by considering $\langle C_3, C_4 \rangle$.

This means that even if a set appears to be a minimal dependent set with respect to a given pair of known circuits, it could be non-minimal with respect to another (potentially yet to be identified) pair of circuits. Similarly, a set may appear independent with respect to a given pair of known circuits but dependent with respect to another set of known circuits. Therefore, in addition to generating new circuit candidates at each iteration, we also need to define the process of eliminating the sets that are proven not to be circuits. This will overcomplicate the algorithm and we decided not to pursue it further. While insufficient by itself, this intermediate step helped us to develop the sufficient conditions for matroid invariance that were presented in Section 6.2.1.

Chapter 7

Conclusion

This thesis is part of the ongoing work to develop and improve efficient schemes for coded private information retrieval. Such information retrieval allows users to download a file from a coded distributed storage system without revealing the identity of the file.

This concept is motivated by the scenario when a collection of files is stored on a distributed storage system and servers send honest responses, but some servers form groups where every member shares the information about incoming queries it observes. Furthermore, we are interested in information-theoretic privacy where the queries do not provide any information about the identity of the file of interest.

Groups of servers that share information among themselves are called colluding sets and their ensemble is a collusion pattern. We worked under the following assumptions:

1. once a group of servers colludes, then every server in the group knows exactly which queries every other member of the group receives, *i.e.*, collusion is closed under containment;
2. due to external constraints, a group of colluding servers cannot pass information to other groups, *i.e.*, collusion is not transitive;
3. there exists at least one non-empty colluding set.

Mathematically, a distributed storage system corresponds to a coordinate space C over $\mathbb{F}_q$ and a collusion pattern to an abstract simplicial complex $\mathcal{T}$ with every facet corresponding to an inclusion maximal colluding set.

We explored the properties of the star-product scheme [10] under arbitrary collusion patterns. This scheme requires the user to choose a publicly known retrieval code Q, which is a vector space over $\mathbb{F}_q$ on the same coordinate set as C.

We achieve privacy with the star-product scheme by constructing an auxiliary code $Q^{\mathcal{T}}$ that is indistinguishable from Q for the colluding servers and call it the lift of Q over a collusion pattern $\mathcal{T}$. Mathematically, for each $T \in \mathcal{T}$ we choose an inclusion maximal linear code Q^T such that $Q_{|T} = Q^T_{|T}$ and then $Q^{\mathcal{T}} = \bigcap_{T \in \mathcal{T}} Q^T$. We can assume that there is no collusion across the entire storage system, for, in this case, the only way to preserve privacy is to download the entire collection of files. In our thesis, we provided an algorithm for constructing the lift for a given pair of a retrieval code Q and an arbitrary collusion pattern $\mathcal{T}$.

A good retrieval code Q allows to download files privately and maximizes the amount of retrieved information in each iteration of the scheme. In general, the choice depends both on the combinatorics of the collusion pattern and the algebraic properties of the storage and retrieval codes. We focused on the former and showed that the operation of lifting is not matroid invariant in full generality. This means that even if two codes Q_1 and Q_2 have the same matroid $M(Q_1) = M(Q_2)$, then their lifts $Q_1^{\mathcal{T}}$ and $Q_2^{\mathcal{T}}$ over a given collusion pattern $\mathcal{T}$ can have different matroids. However, we also provided sufficient conditions under which $M(Q_1^{\mathcal{T}}) = M(Q_2^{\mathcal{T}})$.

This groundwork offers a starting point for several research directions. First of all, we would like to give the necessary and sufficient conditions under which the lifts of two different codes have the same matroid. This will help explain the conditions on the collusion pattern under which the lift of the code is equal to the code itself, *i.e.*, when it is not possible to retrieve data using the star product scheme. Furthermore, it will be interesting to look at a fixed class of retrieval codes, *e.g.*, Reed–Solomon codes, and define a function that gives the dimension of the lift over a given collusion pattern. Similarly, we could to fix a certain class of collusion patterns and see the implications for the dimension of the lift. Possible areas of focus are the lifts that can be interpreted as interesting classes of graphs, *e.g.*, clique complexes or neighborhood complexes.

Bibliography

[1] AL-Ashker, M. M. Coding theory lectures, 2010. [Online] Available: http://site.iugaza.edu.ps/mashker/files/coding-theory-lecturs-for-m-Al-Ashker.pdf. [Accessed: 31- May- 2019]. Islamic University of Gaza.

[2] Banawan, K., and Ulukus, S. Private information retrieval from coded databases. In *2017 IEEE International Conference on Communications (ICC)* (May 2017), pp. 1–6.

[3] Banawan, K., and Ulukus, S. The capacity of private information retrieval from coded databases. *IEEE Transactions on Information Theory 64*, 3 (2018), 1945–1956.

[4] Blackburn, S. R., Etzion, T., and Paterson, M. B. PIR schemes with small download complexity and low storage requirements. In *Information Theory (ISIT), 2017 IEEE International Symposium on* (2017), IEEE, pp. 146–150.

[5] Blahut, R. E. *Theory and Practice of Error Control Codes.* Addison-Wesley, 1983.

[6] Cameron, P. J. Notes on matroids and codes, 1998. [Online] Available: http://www.maths.qmul.ac.uk/~pjc/comb/matroid.pdf. [Accessed: 31- May- 2019]. Queen Mary University of London.

[7] Chor, B., Goldreich, O., Kushilevitz, E., and Sudan, M. Private information retrieval. In *Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on* (1995), IEEE, pp. 41–50.

[8] Chor, B., Goldreich, O., Kushilevitz, E., and Sudan, M. Private information retrieval. *Journal of the ACM (JACM) 45* (1998), 965–981.

[9] Freij-Hollanti, R., Gnilke, O. W., Hollanti, C., Horlemann-Trautmann, A.-L., Karpuk, D., and Kubjas, I. t-private information retrieval schemes using transitive codes. *IEEE Transactions on Information Theory* (2018).

[10] Freij-Hollanti, R., Gnilke, O. W., Hollanti, C., and Karpuk, D. A. Private information retrieval from coded databases with colluding servers. *SIAM Journal on Applied Algebra and Geometry* (2017).

[11] Gordon, G., and McNulty, J. *Matroids: A Geometric Introduction.* Cambridge University Press, 2012.

[12] Guruswami, V. Notes 6: Reed–Solomon, BCH, Reed–Muller, and concatenated codes. Introduction to Coding Theory, 2010. [Online] Available: `https://www.cs.cmu.edu/~venkatg/teaching/codingtheory/notes/notes6.pdf`. [Accessed: 31- May- 2019]. Carnegie Mellon University.

[13] Hall, J. I. Chapter 5: Generalized Reed–Solomon Codes, 2015. [Online] Available: `https://users.math.msu.edu/users/jhall/classes/codenotes/GRS.pdf` [Accessed: 31- May- 2019]. Michigan State University.

[14] Holzbaur, L., Freij-Hollanti, R., and Hollanti, C. On the capacity of private information retrieval from coded, colluding, and adversarial servers. *arXiv preprint arXiv:1903.12552* (2019).

[15] Johnson, W. Matroids, 2009. [Online] Available: `https://www.math.washington.edu/~morrow/336_09/papers/Will.pdf`. [Accessed: 31- May- 2019]. University of Washington.

[16] Kushilevitz, E., and Ostrovsky, R. Replication is not needed: Single database, computationally-private information retrieval. In *Foundations of Computer Science, 1997. Proceedings., 38th Annual Symposium on* (1997), IEEE, pp. 364–373.

[17] Lavauzelle, J., Tajeddine, R., Freij-Hollanti, R., and Hollanti, C. Private information retrieval schemes with regenerating codes. *arXiv preprint arXiv:1811.02898* (2018).

[18] Lee, J. *Introduction to topological manifolds*, vol. 202. Springer Science & Business Media, 2010.

[19] Mirandola, D., and Zémor, G. Critical pairs for the product Singleton bound. *IEEE Transactions on Information Theory 61*, 9 (Sept 2015), 4928–4937.

[20] Moorhouse, E. Notes: Reed–Solomon Codes, 2008. [Online] Available: `http://ericmoorhouse.org/handouts/reed_solomon.pdf`. [Accessed: 31- May- 2019]. University of Wyoming.

[21] Munkres, J. R. *Elements of algebraic topology.* Addison-Wesley Publishing Company, 1984.

[22] Randriambololona, H. An upper bound of singleton type for componentwise products of linear codes. *IEEE Trans. Information Theory 59*, 12 (2013), 7936–7939.

[23] Randriambololona, H. On products and powers of linear codes under componentwise multiplication. *Algorithmic arithmetic, geometry, and coding theory. Contemp. Math 637* (2015), 3–78.

[24] Shah, N. B., Rashmi, K., and Ramchandran, K. One extra bit of download ensures perfectly private information retrieval. In *Information Theory (ISIT), 2014 IEEE International Symposium on* (2014), IEEE, pp. 856–860.

[25] Shannon, C. E. A mathematical theory of communication. *Bell system technical journal 27*, 3 (1948), 379–423.

[26] Shokrollahi, A. Lecture 4: Evaluation codes. Modern coding theory, 2008-2009. [Online] Available: `https://algo.epfl.ch/_media/en/courses/2008-2009/mct_l04.pdf`. [Accessed: 31- May- 2019]. EPFL.

[27] Sun, H., and Jafar, S. A. The capacity of symmetric private information retrieval. *IEEE Transactions on information theory abs/1606.08828* (2019).

[28] Tajeddine, R., Gnilke, O. W., and El Rouayheb, S. Private information retrieval from MDS coded data in distributed storage systems. *IEEE Transactions on Information Theory* (2018).

[29] Tajeddine, R., Gnilke, O. W., Karpuk, D., Freij-Hollanti, R., and Hollanti, C. Private information retrieval from coded storage systems with colluding, byzantine, and unresponsive servers. *IEEE Transactions on information theory* (2019).

[30] Tajeddine, R., Gnilke, O. W., Karpuk, D. A., Freij-Hollanti, R., Hollanti, C., and Rouayheb, S. E. Private informa-

tion retrieval schemes for coded data with arbitrary collusion patterns. *IEEE International Symposium on Information Theory (ISIT) abs/1701.07636* (2017).

[31] Van Lint, J. H. *Introduction to coding theory*, vol. 86. Springer, 1999.

[32] Vu, M. Lecture 1: Entropy and mutual information, 2015. [Online] Available: `http://www.ece.tufts.edu/ee/194NIT/lect01.pdf`. [Accessed: 31- May- 2019]. Tufts University Electrical and Computer Engineering.

[33] Wang, Y. Chapter 2: Simplicial Complex Topics in Computational Topology: An Algorithmic View, 2018. [Online] Available: `http://web.cse.ohio-state.edu/~wang.1016/courses/5559/Lecs/Chap2-complexes.pdf` [Accessed: 31- May- 2019]. Ohio State University.

[34] Westerbäck, T. Matroid Theory. Lecture 4, 2018. [Online] Available: `https://mycourses.aalto.fi/mod/resource/view.php?id=166556`. [Accessed: 31- May- 2019]. Aalto University.

Appendix A

Download rate

One may be interested in understanding the download rate of the star-product PIR scheme. In our case, this is the number of information symbols that can be downloaded with the help of the lift and is determined by $\dim(C \star Q^{\mathcal{T}} / C \star Q)$. However, when the storage code C is a repetition code, this is not an interesting question since downloading any one symbol privately is sufficient to reconstruct the original file. On the other hand, if C is not a repetition code, then the download rate is not determined by the retrieval code Q alone because $\dim(C \star Q^{\mathcal{T}} / C \star Q) \leq \dim(Q^{\mathcal{T}}/Q)$ and equality needs not to hold in general.

Nevertheless, $\dim(Q^{\mathcal{T}}/Q)$ provides a useful upper bound, as it gives the total number of symbols that is possible to retrieve from one row per iteration. Recall that, for a given collusion pattern $\mathcal{T}$, we denote $\dim(Q^{\mathcal{T}}/Q)$ by $\rho(\mathcal{T}, Q)$. Since Q is a vector space, we know that $\rho(\mathcal{T}, Q) > 0$ if and only if $Q^{\mathcal{T}} \backslash Q \neq \emptyset$. Also, $\rho(\mathcal{T}, D) \leq |E| - k$ by the rank-nullity theorem and since $D^{\mathcal{T}} \subset \mathbb{F}_q^n$.

If S is a subset of the ground set E and a new code is formed by puncturing in the coordinates of $E - S$, we denote the dimension of $\dim(Q^{\mathcal{T}}_{|S}/Q_{|S})$ by $\rho_{|S}(\mathcal{T}, Q)$.

Proposition 8. *Let $S \subseteq E$. Then $\rho_{|S}(\mathcal{T}, Q) = \max d_{|S}(\mathbf{x}, Q)$ for $\mathbf{x} \in Q^{\mathcal{T}} - Q$.*

Proof. By definition, if $\mathbf{x} \in Q$, then $\mathbf{x} \equiv \mathbf{0}$ in $Q^{\mathcal{T}}/Q$. Also, it follows from the basic properties of Q^T discussed in Section 6.1 that if a set S is dependent in $Q^{\mathcal{T}}$, then it is dependent in Q. Hence, $\rho(\mathcal{T}, S)$ is the largest number of coordinates of S where a word can take arbitrary values in $Q^{\mathcal{T}}$ but not in Q.

Consider a codeword $\mathbf{x} = (x_1, \cdots, x_n) \in Q^{\mathcal{T}}$. Then $\mathbf{x}$ can be written as $\mathbf{x} = \mathbf{x}' + \mathbf{y}$, where $y_i = x_i' - x_i$ and $\mathbf{x}' \in Q$. In general, the choice of $\mathbf{x}'$ is not unique, however, $\rho_{|S}(\mathcal{T}, Q) = \max w(\mathbf{y}) = \max d_{|S}(Q^{\mathcal{T}} - Q, Q)$. □

In other words, given an arbitrary string of length $|S| \leq n$, we select the longest subset of the string that could be part of a legitimate codeword in Q. The remaining coordinates are deviations from Q. By construction, $Q^{\mathcal{T}}$ maximizes the number of coordinates where the codewords can deviate under the constraints induced by the collusion pattern $\mathcal{T}$. Then $\rho_{|S}(\mathcal{T}, Q)$ measures precisely the number of such coordinates.

Generally speaking, the concept of download rate can have more interpretations that the one that has been given in this thesis. We will offer three most natural options.

First of all, one could talk about the **symbol rate** R_S, which is the highest achievable ratio of the number of downloaded symbols to the weight of the response vector during an iteration of a scheme. Another option is the **file rate** R_F, which gives the highest rate at which one can download the entire file and is the one that is used in our work. Finally, one may be interested in the **stripe rate** R_R, which shows how efficiently the rows (stripes) of the file can be downloaded individually.

It holds that

$$R_R \leq R_F \leq R_S.$$

The first inequality holds because a file can be always downloaded one row at a time. The second inequality holds because some collusion patterns may allow for high rates during the iterations where few information symbols are downloaded.