Graph-based Locally Repairable Codes

Jung-Hyun Kim

The Graduate School

Yonsei University

Department of Electrical and Electronic Engineering

Graph-based Locally Repairable Codes

A Dissertation

Submitted to the Department of

Electrical and Electronic Engineering

and the Graduate School of Yonsei University

in Partial Fulfillment of the Requirements

for the Degree of Doctor of Philosophy

in Electrical and Electronic Engineering

Jung-Hyun Kim

June 2017

This certifies that the dissertation of
Jung-Hyun Kim is approved.

Song Hong Yeop

Thesis Supervisor: Hong-Yeop Song

Dong Ku Kim

Dong Ku Kim

Chungyong Lee

Chungyong Lee

Sooyong Choi

Sooyong Choi

Jon-Lark Kim

The Graduate School
Yonsei University
June 2017

감사의 글

평온했던 에트리에서의 연구원 생활을 뒤로하고 많은 우려 속에 시작된 박사 생활이 어느덧 끝을 향해 달려가고 있습니다. 의욕 넘치게 수업에 참여했던 순간들, 풀리지 않는 문제에 좌절했던 순간들, 미국 학회에서 수상하며 가슴 벅찼던 순간들, 강의와 연구를 병행하며 지치고 힘들었던 순간들이 모두 주마등처럼 지나갑니다. 4년이 넘는 긴 시간동안 이 모든 순간들을 곁에서 믿음으로 함께해준 사랑하는 아내 새봄이에게 가장 먼저 고맙다는 말을 전하고 싶습니다.

늘 든든한 격려와 응원으로 어두운 길을 밝혀주신 순천 부모님과 대전 부모님께 가슴 깊이 감사드립니다. 바쁜 가운데 매번 자신의 일처럼 걱정해주고 기뻐해준 누나와 매형, 늘 내 편이 되어주는 처제에게도 감사와 고마움을 전합니다. 멀리서 기도로 또 가까이서 조언으로 큰 힘이 되어주신 친척들, 친구들, 그리고 선후배님들 모두에게 감사의 마음을 전합니다.

학문에 임하는 자세와 열정을 깨우치게 해주신 송홍엽 교수님, 아낌없는 조언과 격려로 보다 나은 논문이 될 수 있도록 지도해주신 김동구 교수님, 이충용 교수님, 최수용 교수님, 그리고 서강대 수학과 김종락 교수님께도 감사드립니다.

끝으로 제가 사랑하고 존경하는 모든 분들과 저를 아껴주고 믿어주는 모든 분들께 감사의 마음과 함께 자그마한 정성의 결실이 담긴 이 논문을 바칩니다.

2017년 6월

김정현 드림

Contents

List of Figures

List of Tables

ABSTRACT

Graph-based Locally Repairable Codes

Jung-Hyun Kim
Department of Electrical
and Electronic Engineering
The Graduate School
Yonsei University

In this dissertation, we investigate locally repairable codes (LRCs) which recover any decodable set of erased symbols from any single set of disjoint other symbol sets of small size. In particular, if an LRC has a property that an erased symbol can be recovered from any single set of t disjoint sets of other symbols, each set size at most r, it is said to have locality r and availability t.

First, we propose a new graph-based code construction for LRCs having good locality and availability properties. To obtain such codes, we initially use well-known graphs, complete graphs and complete multipartite graphs, and newly defined graphs, each named Tiara graphs, Crown graphs, and Ring graphs. From the simple graphs, we design binary LRCs with good joint locality. Then, we generalize the construction using linear r-uniform t-regular hypergraphs. From the hypergraphs, we design binary LRCs with locality r and availability t. These codes are optimal in terms of a well-known Singleton-like bound by Rawat et al. 2016. If we design such an LRC, we can also obtain a corresponding hypergraph. Thus, our results may open new perspectives on the relation between LRCs and hypergraphs.

Secondly, we extend the graph-based LRCs into block-punctured Simplex codes. We named them Combination codes. The Combination codes and punctured Combination codes are the best among existing codes in terms of the bound by Tamo et al. 2016.

Finally, we define new generalized notions of availability, availability for multiple symbols and joint availability. Then, we derive two new alphabet-dependent bounds for LRCs with joint information availability. The first one is for the code dimension, and the second one is for the minimum distance. Our bounds generalize three recent alphabet-dependent bounds, each by Cadambe et al. 2015, Kim et al. 2016, and Huang et al. 2016.

Key words : Locally repairable codes, Graph-based codes, Hypergraphs, Punctured codes, Alphabet-dependent bounds.

Chapter 1

Introduction

1.1 Background

In the *Big Data* era, due to the dramatically boost in data, distributed storage systems (a.k.a. cloud storage networks) are becoming progressively more important. To guarantee the reliability against storage node failures, various coding techniques have been applied to the systems. The simplest and most commonly used way is replication, where every node is replicated several times, usually three. Such systems are very easy to implement, but severely inefficient in storage space, equipments, devices, and cost for powering and cooling. On the opposite side, maximum distance separable (MDS) codes [1] have minimal storage overhead for a given reliability requirement, but suffer from inefficiency in the repair process.

Motivated by the desire to reduce repair cost of codes for distributed storage systems, an interesting notion, locality, was introduced in the literature [2]. More precisely, if a symbol of a code $\mathcal{C}$ can be expressed as a function of at most r other symbols in $\mathcal{C}$, the symbol is said to have locality r. It is said that a code $\mathcal{C}$ has all symbol locality r if every symbol in $\mathcal{C}$ has the locality at most r. For a systematic code, it is said that the code has information symbol locality r if its every information symbol has the locality at most r. We call such codes locally repairable codes (LRCs) [5]. Some bounds for LRCs have been reported in [2–4], and some LRC constructions have also been proposed in [5–8].

However, for the codes with locality r, a problem arises when multiple node failures occur in the systems. To overcome this problem, the authors of [9] presented the notion of (r, δ)-locality. A symbol of a code $\mathcal{C}$ is said to have (r, δ)-locality, if there exists a punctured subcode of $\mathcal{C}$ which contains the symbol with the length at most $r + \delta - 1$ and minimum distance δ. We refer a systematic code to an LRC with $(r, \delta)_i$-locality if its every information symbol has at most (r, δ)-locality and a code to an LRC with $(r, \delta)_a$-locality if its every symbol has at most (r, δ)-locality. An LRC with $(r, \delta)_a$-locality tolerates up to $\delta - 1$ failures in any local repair process. A bound for LRCs with $(r, \delta)_a$-locality is also presented in [9]. In terms of the bound, the existence of optimal LRCs and some constructions of them are investigated in [10].

Recently, the authors of [11] proposed a generalized notion of locality for multiple node failures, (r_l, l)-cooperative locality: any set of l symbols are expressed as functions of at most r_l other symbols. In this paper, we call this l-locality r_l briefly. The authors in [12–14] suggested joint locality which consider multiple values of l instead of a single value of l. For example, when we consider $l = 1$ and $l = 2$ together, the joint locality is represented by (r_1, r_2).

In addition to the locality, the availability was introduced in [15] as another important property of LRCs. A symbol of a code is said to have (r, t)-availability if it can be recovered from any single set of t disjoint repair sets of other symbols, each set of size at most r. We refer a systematic code to an LRC with $(r, t)_i$-availability if its every information symbol has the locality at most r and availability at least t and a code to an LRC with $(r,t)_a$-availability if its every symbol has the locality at most r and availability at least t. An LRC with $(r,t)_a$-availability also tolerates multiple node failures up to t failures in any local repair process. Moreover, such LRCs ensure parallel reads for each symbol, which is appealing in distributed storage systems containing so-called *hot data* that is frequently and simultaneously accessed by many users. Some bounds for LRCs with availability have been reported in [15–17], and some constructions for optimal LRCs with availability have also been proposed in [15–17].

1.2 Contributions and Organization

In this dissertation, we mainly focus on how to construct LRCs with good locality and availability properties. In Chapter 2, we first introduce the definitions of locality and availability for LRCs. In Chapter 3, we investigate a graph-based code construction for LRCs having good locality and availability properties. To design LRCs with $(r,t)_i$-availability, we use linear r-uniform hypergraphs with vertices of degree at least t. In particular, we can design optimal LRCs with $(r,t)_i$-availability from linear r-uniform t-regular hypergraphs in terms of a well-known Singleton-like bound and a code rate bound implied by it. We present the necessary conditions for the existence of such a hypergraph, that is, an optimal code. Moreover, as a special case, we can also design optimal LRCs with $(r,t)_a$-availability from labelled linear r-uniform t-regular hypergraphs. In Chapter 4, we extend the graph-based LRCs into block-punctured Simplex codes. The block-punctured Simplex codes (also called Combination codes) have the generator matrix with all the columns of weights larger than a given value punctured. We call by block-puncturing such puncturing method. We determine the minimum distance, locality, joint information locality, and availability of the Combination codes in closed-form expressions. To increase the code rate more, we suggest a heuristic puncturing method, called subblock-puncturing, that punctures a few more columns of the largest weight from the generator matrix of the Combination code. Using well-known bounds, we demonstrate the optimality of Combination codes and punctured Combination codes. In Chapter 5, we extend the original availability for one symbol into the availability of multiple symbols, and then, define joint availability of codes. Based on the new notion of availability, we propose two alphabet-dependent bounds. Interestingly, it is possible to deduce some well-known bounds for LRCs from the proposed bounds. Finally, we show the achievability and tightness of the proposed bounds using some graph-based codes and Combination codes.

Chapter 2

Basic Concepts

2.1 Classical Coding Theory

In communication systems, channel coding is a technique to control errors in data transmission over noisy or unreliable communication channels. The main idea is the sender encodes the information data in a redundant way by using an error-correcting code (also simply code). A code can be thought of as a collection of codewords over a finite (Galois) field. We denote by $\mathbb{F}_q$ a finite field of size q.

An $(n, M, d)_q$ code $\mathcal{C}$ is a collection of M codewords of length n over a field $\mathbb{F}_q$. A codeword $c \in \mathcal{C}$ consists of n symbols. The minimum distance d of $\mathcal{C}$ is defined as the minimum Hamming distance between any two distinct codewords in $\mathcal{C}$, or equivalently, the minimum Hamming weight of a codeword in $\mathcal{C}$. The Hamming distance is the number of coordinates on which two codewords differ, and the Hamming weight is the number of non-zero coordinates on a codeword.

A code $\mathcal{C}$ is called linear if $\mathcal{C}$ is a linear subspace of $\mathbb{F}_q^n$. More specifically, for every two codewords $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C}$, we have $a_1\mathbf{c}_1 + a_2\mathbf{c}_2 \in \mathcal{C}$ with two scalars $a_1, a_2 \in \mathbb{F}_q$. If k is the dimension of $\mathcal{C}$, then $k = \log_q M$. In the rest of the paper, we use the notation

$(n, k, d)_q$ to denote parameters of a code of length n, cardinality q^k, i.e., dimension k, minimum distance d over $\mathbb{F}_q$, for both linear and non-linear codes. Similarly, we also use the notation $[n, k, d]_q$ for only linear codes.

Let $\mathcal{C}$ be an $[n, k, d]_q$ code and G be a generator matrix of $\mathcal{C}$. The generator matrix G is a $k \times n$ matrix whose rows form a basis of $\mathcal{C}$. Then, we can encode information words $\mathbf{u} \in \mathbb{F}_q^k$ to coded words (also codewords) of $\mathcal{C}$ by using a mapping $\mathbb{F}_q^k \to \mathbb{F}_q^n$ defined by

$$\mathbf{u} \mapsto \mathbf{u}G.$$

Also, we can apply elementary operations to a generator matrix so that it contains a $k \times k$ identity matrix as a submatrix. The generator matrix is called systematic.

A parity-check matrix H of an $[n, k, d]_q$ code is a $(n-k) \times n$ matrix such that

$$\mathbf{c} \in \mathcal{C} \Leftrightarrow H\mathbf{c}^T = 0,$$

where $(\cdot)^T$ stands for transposition.

From the well-known relationship between the rank of a matrix and the dimension of its kernel, we get

$$HG^T = GH^T = 0.$$

Thus, if a generator matrix G has the form $(I|A)$, we can take the $(n-k) \times n$ matrix $H = (-A^T|I)$ as a parity-check matrix.

2.2 Codes with Locality and Availability

We introduce the locality through a binary Simplex code. Consider a $[7, 3, 4]_2$ Simplex code $\mathcal{C}$. Let G be the generator matrix of $\mathcal{C}$. Consider a vector of information symbols $\mathbf{x} = (x_1, x_2, x_3)$ and its corresponding codeword $\mathbf{c} = (c_1, c_2, \ldots, c_n)$. Then, we have

$$
\begin{aligned}
\mathbf{x} \cdot G &= \begin{pmatrix} x_1 & x_2 & x_3 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \\
&= \begin{pmatrix} x_1 & x_2 & x_3 & x_1 + x_2 & x_1 + x_3 & x_2 + x_3 & x_1 + x_2 + x_3 \end{pmatrix} \\
&= \begin{pmatrix} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 \end{pmatrix} = \mathbf{c}.
\end{aligned}
$$

If we denote by $\mathcal{R}_i(c_1)$, the i-th set of indices of coded symbols which repair c_1, then we have the following:

$$\mathcal{R}_1(c_1) = \{2, 4\}, \quad \mathcal{R}_2(c_1) = \{3, 5\}, \quad \mathcal{R}_3(c_1) = \{6, 7\}. \tag{2.1}$$

$\mathcal{R}_1(c_1)$, $\mathcal{R}_2(c_1)$, and $\mathcal{R}_3(c_1)$ are referred to as repair sets of c_1. The maximum size of any repair set is defined as the locality of the symbol. Thus, the locality of c_1 is 2. The availability of a symbol is defined as the number of disjoint repair sets, each set of size at most the locality value. Thus, the availability of c_1 is 3 since $\mathcal{R}_1(c_1)$, $\mathcal{R}_2(c_1)$, and $\mathcal{R}_3(c_1)$ are disjoint and $|\mathcal{R}_1(c_1)| = |\mathcal{R}_2(c_1)| = |\mathcal{R}_3(c_1)| = 2$. The locality of a code is the maximum value of localities of the symbols, and the availability of a code is the minimum value of availabilities of the symbols.

2.3 From Error-Correcting Codes to Locally Repairable Codes (LRCs)

Consider a $[14, 10, 5]_q$ Reed-Solomon (RS) code which is used in Facebook and many other Hadoop MapReduce architectures [18]. Since it is an MDS code, it has high reliability for a given code rate. However, it always needs 10 symbols to repair only single erasure. To increase the repair efficiency, we use additional local parities.

In Figure 2.1, we add two local parities c_{15} and c_{16}. By adding the local parity $c_{15} = a_1c_1 + a_2c_2 + a_3c_3 + a_4c_4 + a_5c_5$, a single erasure in information symbols can be repaired from only 5 other symbols. For example, if c_3 is lost, it can be reconstructed by $c_3 = a_c^{-1}(c_{15} - a_1c_1 - a_2c_2 - a_4c_4 - a_5c_5)$. The coefficients $a_1, a_2, \ldots, a_{10}$ should be chosen so that the parities satisfy an equation $a_{11}c_{11} + a_{12}c_{12} + a_{13}c_{13} + a_{14}c_{14} + a_{15}c_{15} + a_{16}c_{16} = 0$. When a single erasure in parity symbols happens, the equation can be used to repair the symbol. For example, if c_{12} is lost, it can be recovered by $c_{12} = a_{12}^{-1}(-a_{11}c_{11} - a_{13}c_{13} - a_{14}c_{14} - a_{15}c_{15} - a_{16}c_{16})$.

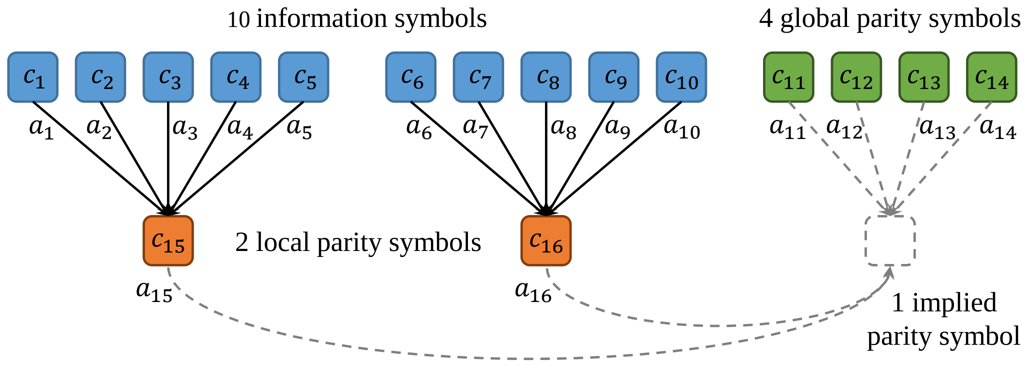


Figure 2.1: Description of an LRC with locality 5.

Figure 2.2 shows how to construct a generator matrix G' for an LRC with (r, t)-availability from a generator matrix G of a systematic MDS code. The first $k + m$ columns of G' are the same with the first $k + m$ columns of G. The last $t\frac{k}{r}$ columns of G' are obtained by splitting each of the last t columns of G into $\frac{k}{r}$ columns of weight r each. A detailed description of the construction is given in [15].

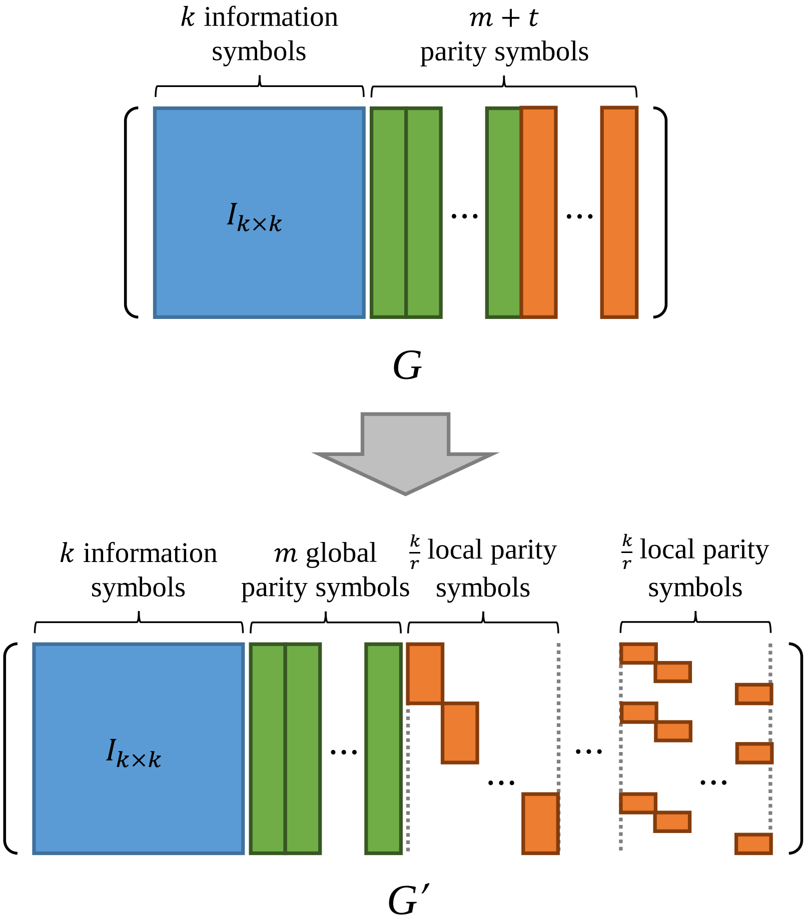


Figure 2.2: Description of the construction of a generator matrix G' for an LRC with (r, t)-availability from a generator matrix G of a systematic MDS code.

Chapter 3

Graph-based Locally Repairable Codes

3.1 Graph-based LRCs

Interpreting codes as graphs and graphs as codes opens new perspectives for constructing such error-correcting codes. Low-density parity-check (LDPC) code [19] is one of the most recent examples of codes defined on graphs. An LDPC code can be represented by a bipartite graph (a.k.a. Tanner graph) made up of two disjoint vertices, variable vertices and check vertices. The variable vertices represent coded symbols including information symbols and parity symbols. The check vertices represent parity check equations. An edge connects a variable vertex to the check vertex if and only if the coded symbol is included in the parity check equation. From the graph, we can obtain a parity check matrix for an LDPC code.

There are several different ways to represent a graph by a code, but the one we favor in this paper is as follows. Consider a graph with vertices and edges. In the graph, the vertices are associated with information symbols, and the edges are associated with parity symbols. Each parity symbol is computed by addition of information symbols joined to the parity symbol. Then, we can convert the graph into a code. More precisely, from the graph, we can obtain a generator matrix for a code. In this chapter, we use this approach and generalize it using hypergraphs [20] to design LRCs. By designing a generator matrix of a code using hypergraph, we can intuitively check the locality and availability of the code and obtain its minimum distance. Moreover, we can use some already known results related to hypergraph such as existence or equivalence relations.

3.1.1 LRCs from Graphs

We first suggest a graph-based code construction for binary LRCs using simple graphs. This construction provides binary LRCs with $(r = 2, t)$-availability.

Definition 1 Consider a simple graph, that is an unweighted, undirected graph containing no graph loops or multiple edges. In the graph, the vertices are associated with information symbols, and the edges are associated with parity symbols. Each parity symbol is calculated by the summation of adjacent information symbols. Then, we can convert the graph into a code $\mathcal{C}$. We call this code $\mathcal{C}$ graph-based code. □

Theorem 1 Let $\mathcal{C}$ be a graph-based code. The minimum distance d of $\mathcal{C}$ satisfies

$$d = \min_{\Upsilon \subseteq V} \left[|Cut(\Upsilon, \Upsilon^c)| + |\Upsilon| \right], \tag{3.1}$$

where V is the vertex set. Υ and Υ^c are a subset of V and its complementary set in V, respectively. The $Cut(\Upsilon, \Upsilon^c)$ is a cut which is a partition of V into two disjoint subsets, Υ and Υ^c. The size of $Cut(\Upsilon, \Upsilon^c)$ is the number of edges crossing the cut.

Proof: Consider a subset Υ in a graph representation $\mathcal{G}$ of $\mathcal{C}$. All the edges shared by any vertex pairs in Υ do not affect the size of $Cut(\Upsilon, \Upsilon^c)$. In a standard generator matrix G of $\mathcal{C}$, choose rows corresponding to Υ. We denote this submatrix G_Υ. Now, sum up all the rows in G_Υ by column-wise modulo-2 addition to generate a codeword. The weight of this codeword is the same with the value of $|Cut(\Upsilon, \Upsilon^c)| + |\Upsilon|$. Therefore, for all the possible vertex set Υ, the minimum value of $|Cut(\Upsilon, \Upsilon^c)| + |\Upsilon|$ is the minimum weight of codewords, that is the minimum distance of $\mathcal{C}$. ■

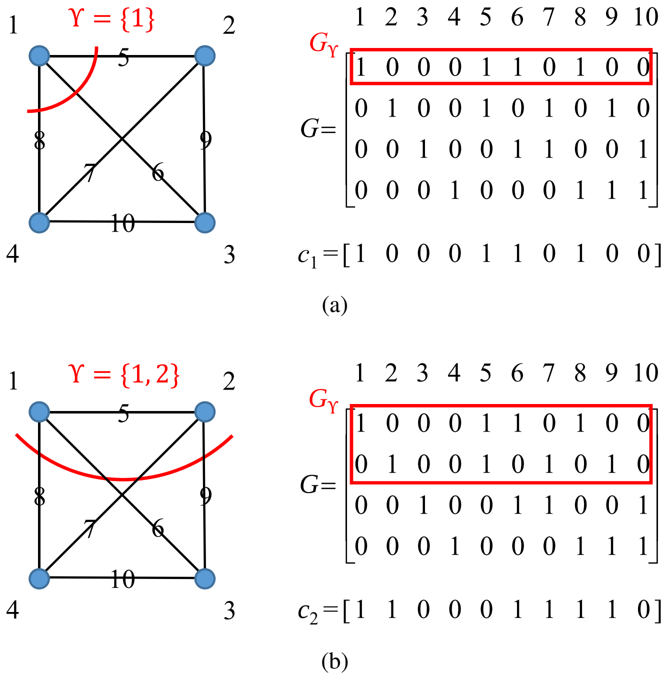


Figure 3.1: Two examples of $Cut(\Upsilon, \Upsilon^c)$. (a) Case 1; (b) Case 2.

Example 1 Consider a graph-based code of dimension 4 in Figure 3.1. Fig. 3.1(a) shows the case of $\Upsilon = \{1\}$. In this case, $|Cut(\Upsilon, \Upsilon^c)| + |\Upsilon| = 4$, and the value is the same with the weight of the codeword c_1. The c_1 is a codeword generated by G_Υ itself. Fig.

3.1(b) shows the case of $\Upsilon = \{1,2\}$. In this case, $|Cut(\Upsilon, \Upsilon^c)| + |\Upsilon| = 6$, and the value is the same with the weight of the codeword c_2. The c_2 is a codeword generated by performing modulo-2 addition to two rows of G_Υ.

Now, to obtain binary LRCs having good joint locality properties, we use two well-known graphs, complete graphs and complete multipartite graphs, and three newly defined graphs, Tiara graphs, Crown graphs, and Ring graphs, as the following.

Definition 2 Consider a complete graph which has k vertices and $\binom{k}{2}$ edges. Then, we can convert the graph into a code $\mathcal{C}$. We call this code $\mathcal{C}$ Complete Graph (CG) code.

Definition 3 Consider a complete p-partite (k/p)-uniform graph with k vertices. For an integer p, $p|k$, partition the vertex set into p subsets. Each subset contains k/p vertices. Connect every two vertices in different subsets with an edge. The number of edges becomes $k(k-k/p)/2$. Then, we can convert the graph into a code $\mathcal{C}$. We call this code $\mathcal{C}$ Complete Multipartite Graph (CMG) code. □

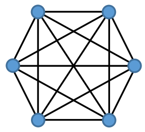
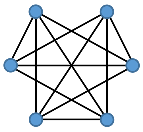
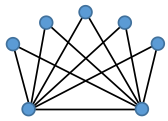
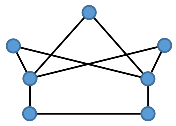
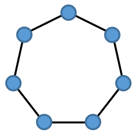
Definition 4 Consider a graph with $k \geqslant 3$ vertices. At first, map any three information symbols onto vertices in a triangle graph, and choose a vertex pair. If there exists the remained information symbols, connect each of them with the chosen vertex pair in the triangle graph by two edges. Then, we can convert the graph into a code $\mathcal{C}$. We call this code $\mathcal{C}$ Tiara code. □

Definition 5 Consider a graph with $k \geqslant 5$ vertices. At first, map any five information symbols onto vertices in a pentagon graph, and choose a non-adjacent vertex pair. If there exists the remained information symbols, connect each of them with the chosen vertex pair in the pentagon graph by two edges. Then, we can convert the graph into a code $\mathcal{C}$. We call this code $\mathcal{C}$ Crown code. □

Definition 6 Consider a cycle graph with $k \geqslant 3$ vertices and k edges. Then, we can convert the graph into a code $\mathcal{C}$. We call this code $\mathcal{C}$ Ring code. □

We note that joint localities of the above graph-based codes are easily obtained by the structures of the corresponding graphs [12–14]. The minimum distance of the codes are obtained by Theorem 1. Table 3.1 summarizes the code parameters, locality properties, graphs, and generator matrices of the above graph-based codes. In the Table, $(\cdot)_i$ and $(\cdot)_a$ denote joint localities for information symbols and all symbols, respectively.

Table 3.1: Graph-based LRCs and their joint localities, graphs, and generator matrices

Graph-based LRCs	Joint Localities	Graphs ($k = 6, 7$)	Generator Matrices ($k = 6, 7$)
$(\frac{k(k+1)}{2}, k, k)_2$ CG code	$(r_1, r_2)_i = (2, 3)$ $(r_1, r_2)_a = (2, 3)$		$\begin{pmatrix} 1&0&0&0&0&0&1&1&0&1&0&0&1&0&0&0&1&0&0&0&0 \\ 0&1&0&0&0&0&1&0&1&0&1&0&0&1&0&0&0&1&0&0&0 \\ 0&0&1&0&0&0&0&1&1&0&0&1&0&0&1&0&0&0&1&0&0 \\ 0&0&0&1&0&0&0&0&0&1&1&1&0&0&0&1&0&0&0&1&0 \\ 0&0&0&0&1&0&0&0&0&0&0&0&1&1&1&1&0&0&0&0&1 \\ 0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&1&1&1&1&1 \end{pmatrix}$
$(\frac{k(k-\frac{k}{p}+2)}{2}, k, k-\frac{k}{p}+1)_2$ CMG code	$(r_1, r_2)_i = (2, 3)$ $(r_1, r_2)_a = (2, 4)$		$\begin{pmatrix} 1&0&0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0&0&0&1&1&1&1&0&0&0&0 \\ 0&0&1&0&0&0&1&0&0&0&1&0&0&0&1&1&0&0 \\ 0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&0&1&1 \\ 0&0&0&0&1&0&0&0&1&0&0&0&1&0&1&0&1&0 \\ 0&0&0&0&0&1&0&0&0&1&0&0&0&1&0&1&0&1 \end{pmatrix}$
$(3k - 3, k, 3)_2$ Tiara code	$(r_1, r_2)_i = (2, 3)$ $(r_1, r_2)_a = (2, 3)$		$\begin{pmatrix} 1&0&0&0&0&0&0&1&1&0&1&0&1&0&1&0&1&0 \\ 0&1&0&0&0&0&0&1&0&1&0&1&0&1&0&1&0&1 \\ 0&0&1&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0&0&0&0&1&1&0&0&0&0 \\ 0&0&0&0&0&1&0&0&0&0&0&0&0&0&1&1&0&0 \\ 0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&1&1 \end{pmatrix}$
$(3k - 5, k, 3)_2$ Crown code	$(r_1, r_2)_i = (2, 3)$ $(r_1, r_2)_a = (2, 4)$		$\begin{pmatrix} 1&0&0&0&0&0&0&1&0&0&0&1&1&0&1&0 \\ 0&1&0&0&0&0&0&1&1&0&0&0&0&1&0&1 \\ 0&0&1&0&0&0&0&0&1&1&0&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0&0&1&1&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0&0&1&1&0&0&0&0 \\ 0&0&0&0&0&1&0&0&0&0&0&0&1&1&0&0 \\ 0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&1 \end{pmatrix}$
$(2k, k, 3)_2$ Ring code	$(r_1, r_2)_i = (2, 4)$ $(r_1, r_2)_a = (2, 4)$		$\begin{pmatrix} 1&0&0&0&0&0&0&1&0&0&0&0&0&1 \\ 0&1&0&0&0&0&0&1&1&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0&1&1&0&0&0&0 \\ 0&0&0&1&0&0&0&0&0&1&1&0&0&0 \\ 0&0&0&0&1&0&0&0&0&0&1&1&0&0 \\ 0&0&0&0&0&1&0&0&0&0&0&1&1&0 \\ 0&0&0&0&0&0&1&0&0&0&0&0&1&1 \end{pmatrix}$

3.1.2 LRCs from Hypergraphs

In the previous subchapter, we constructed binary LRCs from simple graphs [20]. Now, we generalize the construction using linear uniform hypergraphs [20]. A hypergraph is a generalization of a graph in which an edge can join any number of vertices. A simple hypergraph is an unweighted, undirected hypergraph containing no loops or multiple edges. A hypergraph is linear if it is simple and every two edges shares at most one vertex. A r-uniform hypergraph is a hypergraph such that every edge contains r vertices. Thus, a simple graph is just a linear 2-uniform hypergraph.

By designing a generator matrix of an LRC using such a hypergraph, we can find the locality and availability characteristics of the code intuitively and find the minimum distance relatively easily. We first define a hypergraph code, and then determine the code parameters and properties such as the code length, code rate, minimum distance, locality, and availability [21].

Definition 7 Consider a linear r-uniform hypergraph $\mathcal{H}$. In $\mathcal{H}$, the vertices are associated with information symbols, and the edges are associated with parity symbols. Each parity symbol is calculated by binary addition of information symbols contained in the parity symbol. Then, we can convert $\mathcal{H}$ into a code $\mathcal{C}$. We call this code $\mathcal{C}$ an r-uniform hypergraph code.

Consider a linear r-uniform hypergraph $\mathcal{H}$ with v vertices and e edges. Then, the corresponding code $\mathcal{C}$ has the dimension v, length $v+e$, and thus, code rate $\frac{v}{v+e}$. The number of edges e can be represented by using the degree of vertices in $\mathcal{H}$. We denote by τ_i the degree of i-th vertex in ascending order, i.e. $\tau_1 \leqslant \tau_2 \leqslant \ldots \leqslant \tau_v$. Then,

$e = \frac{1}{r}\sum_{i=1}^{v}\tau_i$, and thus, the code length of $\mathcal{C}$ is $v + \frac{1}{r}\sum_{i=1}^{v}\tau_i$.

Lemma 1 Let $\mathcal{C}$ be a $[v + \frac{1}{r}\sum_{i=1}^{v}\tau_i, v, d]_2$ r-uniform hypergraph code, and G be a systematic generator matrix of $\mathcal{C}$. For any s rows in G, we denote by $\mathbf{x}$ a row of them and by $W(s)$ the weight of a row obtained as a result of summing up all the s rows by column-wise binary addition. Then, $W(s)$ is greater than or equal to the weight of $\mathbf{x}$.

Proof: Using column and row permutation, we can always make G to the form G' in Fig. 3.2. In G', the top s rows are the selected rows, and the first row is $\mathbf{x}$. Consider a submatrix consisting of the remaining $s-1$ rows except $\mathbf{x}$. We can divide the submatrix into M_1 and M_0 as shown in the figure. $W(s)$ can be written as follows:

$$W(s) = \begin{cases} wt(\mathbf{x}), & \text{for } s = 1, \\ wt(\mathbf{x}) - W_1(s-1) + W_0(s-1), & \text{for } 2 \leqslant s \leqslant v, \end{cases}$$

where $wt(\mathbf{x})$ is the weight of $\mathbf{x}$. $W_1(s-1)$ and $W_0(s-1)$ are the numbers of columns of odd weight in M_1 and M_0, respectively. Since M_1 contains only disjoint non-zero columns and at least one all-zero column, $W_1(s-1) \leqslant s-1$. Since M_0 contains at least $s-1$ columns of weight one, $W_0(s-1) \geqslant s-1$. Thus, we have $W(s) \geqslant wt(\mathbf{x})$. ■

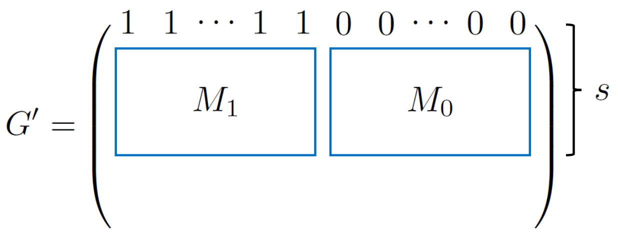


Figure 3.2: A permuted generator matrix of a hypergraph code.

Theorem 2 Let $\mathcal{C}$ be a $[v + \frac{1}{r}\sum_{i=1}^{v}\tau_i, v, d]_2$ r-uniform hypergraph code. Then, the minimum distance d of $\mathcal{C}$ is $\tau_1 + 1$.

Proof: Consider a systematic generator matrix G of $\mathcal{C}$. By Lemma 1, the minimum distance d of $\mathcal{C}$ is the minimum weight of a row in G. Consider a hypergraph $\mathcal{H}$ corresponding to G. The weight of a row in G is one more than the degree of corresponding vertex in $\mathcal{H}$. Thus, we have $d = \tau_1 + 1$. ■

Theorem 3 Let $\mathcal{C}$ be a $[v + \frac{1}{r}\sum_{i=1}^{v}\tau_i, v, d]_2$ r-uniform hypergraph code. Then, $\mathcal{C}$ has $(r, \tau_1)_i$-availability.

Proof: Let $\mathcal{H}$ be a linear r-uniform hypergraph corresponding to $\mathcal{C}$. Every vertex in $\mathcal{H}$ is connected with at least τ_1 edges. Every edge contains r vertices, and every two edges shares at most one vertex. It means that the corresponding information symbol has at least τ_1 disjoint repair sets of cardinality r. ■

Example 2 Consider a 3-uniform hypgergraph $\mathcal{H}$ in Figure 3.3. Let $\mathcal{C}$ be a $[10, 6, d]_2$ code corresponding to $\mathcal{H}$. Based on Theorem 2 and Theorem 3, we can easily check that $\mathcal{C}$ has the minimum distance $d = 3$ and $(3, 2)_i$-availability. Now, let's see these characteristics again using a systematic generator matrix of $\mathcal{C}$. The generator matrix G is given by,

$$G = (I|P) = \begin{pmatrix} 1&0&0&0&0&0&1&1&0&0 \\ 0&1&0&0&0&0&1&0&1&0 \\ 0&0&1&0&0&0&1&0&0&1 \\ 0&0&0&1&0&0&0&1&1&0 \\ 0&0&0&0&1&0&0&1&0&1 \\ 0&0&0&0&0&1&0&0&1&1 \end{pmatrix}.$$

Recall the relation between a generator matrix and its corresponding hypergraph. Since the submatrix P has constant column weight 3 and constant row weight 2, we also check that $\mathcal{C}$ has $(3, 2)_i$-availability from G.

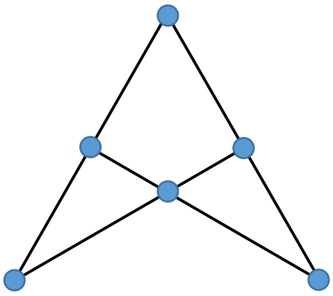


Figure 3.3: A 3-uniform hypergraph.

3.2 Graph-based Optimal LRCs with Availability

For LRCs with $(r,t)_i$-availability, an upper bound on the minimum distance was presented in [15] under the following condition. Let $\mathcal{C}$ be an $[n,k,d]_q$ LRC. Every information symbol of $\mathcal{C}$ has t disjoint repair sets, each set of size at most r, such that any repair set contains only one parity symbol. Then, the minimum distance d of $\mathcal{C}$ is bounded by

$$d \leqslant n - k - \left\lceil \frac{kt}{r} \right\rceil + t + 1. \tag{3.2}$$

Additionally, we note that the bound (3.2) implies

$$\frac{k}{n} \leqslant \frac{r}{r+t}. \tag{3.3}$$

A construction of codes that attain the bound (3.2) with equality is also given in [15]. The key of the construction is how to design suitable binary matrix, called membership matrix, to obtain additional local parities from original global parities of a systematic MDS code. The authors in [15] proposed two explicit designs of the membership matrix based on resolvable designs [22] and zigzag codes [23]. Some other designs of the membership matrix are studied in [24].

For binary codes, from the results of [15], we can obtain Remark 1.

Remark 1 There are four classes of optimal binary LRCs achieving the bound (3.2) as the following. Here, p is a positive integer.

1. For $r \mid kt$, $[k + \frac{kt}{r}, k, t+1]_2$ codes.
2. For $r \mid kt$, $[k + \frac{kt}{r} + p, k, t+1+p]_2$ codes.
3. For $r \nmid kt$, $[k + \lceil \frac{kt}{r} \rceil, k, t+1]_2$ codes.
4. For $r \nmid kt$, $[k + \lceil \frac{kt}{r} \rceil + p, k, t+1+p]_2$ codes.

Recently, the authors in [25] found all the optimal binary LRCs for $t = 1$. The authors in [26] showed that regular LDPC codes with girth greater than 4 can be used to construct optimal codes for the class $1)$. They proposed three constructions from finite geometry LDPC codes [27] and array LDPC codes [28]. Based on the same framework, the authors in [29] proposed two constructions from circulant permutation matrices and affine permutation matrices. The authors in [30] proposed binary LRCs with $(r, t)_a$-availability achieving the bound (3.3) with equality. We can easily check that the codes are a subset of the class $1)$. For $t \geqslant 2$, to the best of our knowledge, any general construction of optimal binary LRCs for the class 2) is not known yet, and the existence of optimal binary LRCs for the classes $3)$ and $4)$ is open. In terms of the bound (3.3), all the codes for the classes $1)$ and $3)$ are optimal but all the codes for the classes 2) and $4)$ are not optimal.

In this chapter, we focus on optimal binary LRCs for the classes $1)$ and $2)$ for $r, t \geqslant 2$. We construct optimal codes for the class $1)$ from linear r-uniform t-regular hypergraphs [20]. In particular, we construct optimal binary LRCs with $(r,t)_a$-availability using specially designed hypergraphs with $v = \binom{r+t-1}{t}$ vertices. We extend some of optimal binary LRCs for the class $1)$ into optimal binary LRCs for the class $2)$.

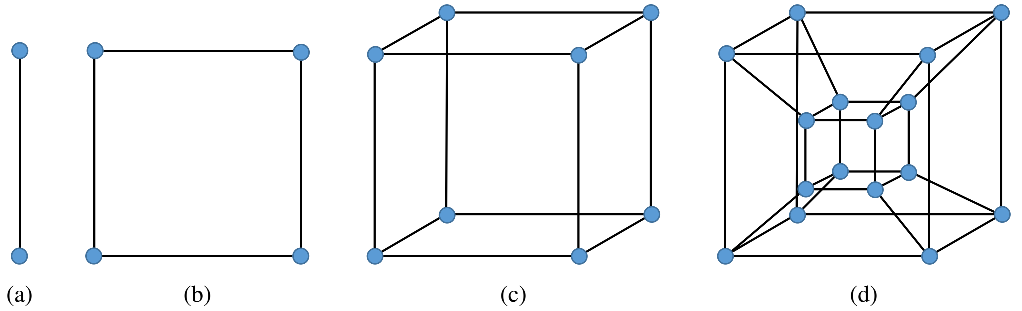


Figure 3.4: Examples of $(2,t)$-hypergraphs ($t = 1,2,3,4$). (a) A $(2,1)$-hypergraph; (b) A $(2,2)$-hypergraph; (c) A $(2,3)$-hypergraph; (d) A $(2,4)$-hypergraph.

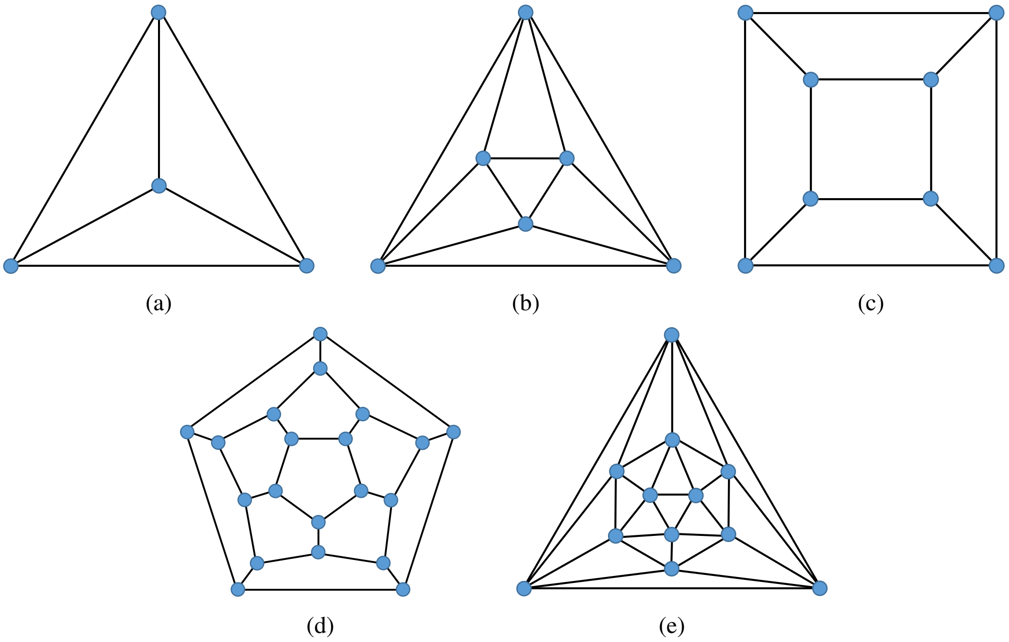


Figure 3.5: Five platonic graphs. (a) Tetrahedron; (b) Octahedron; (c) Hexahedron; (d) Dodecahedron; (e) Icosahedron.

3.2.1 Graph-based Optimal LRCs with Information Symbol Availability

We first consider linear r-uniform t-regular hypergraphs such that every edge contains r vertices, and every vertex is contained in t edges. In the rest of this paper, we refer to a linear r-uniform t-regular hypergraph as an (r, t)-hypergraph. Figure 3.4 shows $(2, t)$-hypergraphs for $t = 1, 2, 3, 4$. The well-known platonic graphs are also examples of $(2, t)$-hypergraphs. Figure 3.5 shows five platonic graphs. Tetrahedron, Hexahedron, and Dodecahedron are $(2, 3)$-hypergraphs. Octahedron is a $(2, 4)$-hypergraph, and Icosahedron is a $(2, 5)$-hypergraph. Figure 3.6 shows an example of another type of (r, t)-hypergraphs. The graph is asymmetric but it is also a $(2, 3)$-hypergraph. Figure 3.7 shows (r, t)-hypergraphs for $(r = 2, t = 4)$, $(r = 3, t = 2)$, and $(r = 3, t = 3)$.

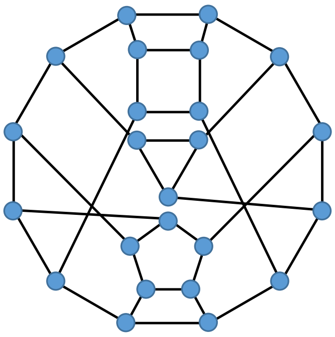
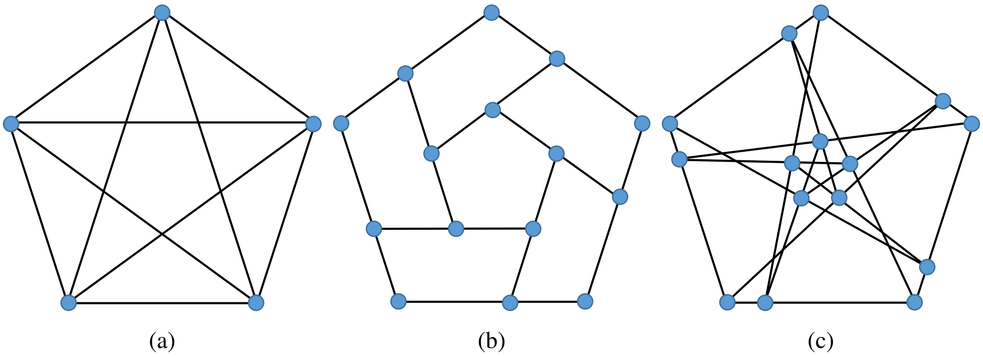


Figure 3.6: An example of asymmetric $(2, 3)$-hypergraphs.

We note that, for any positive integers r and t, the necessary conditions for the existence of an (r, t)-hypergraph on v vertices are $v \geqslant t(r-1)+1$ and $r \mid vt$. For $r = 2, 3$, the necessary conditions are also sufficient. For $r = 4$, no non-existence result of an (r, t)-hypergraph on v vertices is known under the necessary conditions. For $r \geqslant 5$, the necessary conditions are not sufficient. The detailed information of existence under the


(a)

(b)

(c)

Figure 3.7: (r, t)-hypergraphs for $(r = 2, t = 4)$, $(r = 3, t = 2)$, and $(r = 3, t = 3)$. (a) A $(2, 4)$-hypergraph; (b) A $(3, 2)$-hypergraph; (c) A $(3, 3)$-hypergraph.

necessary conditions is presented in [22]. It is also proved that an (r, t)-hypergraph on v vertices always exists with sufficiently large v [31]. One good example is $v = \binom{r+t-1}{t}$. To obtain (r, t)-hypergraphs, we can also use equivalence relations in Remark 2 and useful properties in Remark 3.

Remark 2 [22] Consider an (r, t)-hypergraph $\mathcal{H}$ with v vertices and e edges. Then, the followings are satisfied:

1. $\mathcal{H}$ is a (v_t, e_r)-configuration.

2. $\mathcal{H}$ such that $\lambda(v - 1) = t(r - 1)$ for a positive integer λ is a (v, e, t, r, λ)-design.

3. $\mathcal{H}$ corresponds to a biregular graph with v vertices of degree t, e vertices of degree r, and the girth at least 6. We call this graph a Levi graph.

Remark 3 [22] Consider an (r, t)-hypergraph $\mathcal{H}$ with v vertices and e edges. Then, the followings are satisfied:

1. The dual of $\mathcal{H}$ is a (t, r)-hypergraph with e vertices and v edges.

2. Suppose that we have an (r, t)-hypergraph $\mathcal{H}'$ with v' vertices and e' edges, with $\mathcal{H}$ and $\mathcal{H}'$ being disjoint. Then, we can obtain a new (r, t)-hypergraph $\mathcal{H}''$ with $v'' = v + v'$ vertices and $e'' = e + e'$ edges by combining $\mathcal{H}$ and $\mathcal{H}'$.

Definition 8 Consider an (r, t)-hypergraph $\mathcal{H}$ with v vertices and $\frac{vt}{r}$ edges. In $\mathcal{H}$, the vertices are associated with information symbols, and the edges are associated with parity symbols. Each parity symbol is calculated by binary addition of information symbols contained in the parity symbol. Then, we can convert $\mathcal{H}$ into a code $\mathcal{C}$. We call this code $\mathcal{C}$ an (r, t)-hypergraph code.

We note that a $[v + \frac{vt}{r}, v, d]_2$ (r, t)-hypergraph code has the minimum distance $d = t + 1$ by Theorem 2 and $(r, t)_i$-availability by Theorem 3. With these parameters, we can easily check that (r, t)-hypergraph codes are optimal in terms of the bounds (3.2) and (3.3), and thus, they are optimal codes for the class $1)$ in Remark 1. Moreover, by the definition, the necessary conditions for the existence of (r, t)-hypergraph codes are also those of optimal codes for the class $1)$.

Remark 4 Using the $(2, 4)$-hypergraph in Fig. 3.7(a), we can obtain a $[15, 5, 5]_2$ LRC with $(2, 4)_i$-availability. It can not be obtained from any three constructions in [26] or two constructions in [29]. Moreover, all the codes in [26] and [29] are obtained from our construction, and hence, ours is more general and encompasses those in [26] and [29].

3.2.2 Graph-based Optimal LRCs with All Symbol Availability

As a special case of the graph-based optimal binary LRCs with $(r,t)_i$-availability, we can construct optimal binary LRCs with $(r,t)_a$-availability from specially designed hypergraphs with $v = \binom{r+t-1}{t}$ vertices. We first design such hypergraphs using vertex labelling. We call them (r,t)-tower hypergraphs. The detailed definition is presented in the following.

Definition 9 For given two positive integers r and t, consider a hypergraph $\mathcal{H}$ with v vertices labelled using t-subsets of a $(r+t-1)$-set, that is, $v = \binom{r+t-1}{t}$. Connect a subset of vertices with an edge if and only if all the vertices in the subset share $t-1$ indices, and label the edge using the common indices. We call this hypergraph $\mathcal{H}$ an (r,t)-tower hypergraph.

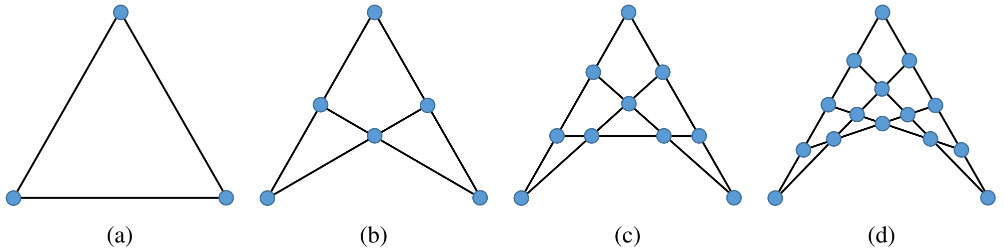


Figure 3.8: (r,t)-tower hypergraphs for $r = 2,3,4,5$ and $t = 2$. (a) A $(2,2)$-tower hypergraph; (b) A $(3,2)$-tower hypergraph; (c) A $(4,2)$-tower hypergraph; (d) A $(5,2)$-tower hypergraph.

Figure 3.8 shows (r,t)-tower hypergraphs for $r = 2,3,4,5$ and $t = 2$ without the labels. Each of them has single tower structure. A tower forms a dependent set for edges contained in the tower. That is, each edge in the tower can be represented by the other edges. For $t \geqslant 3$, an (r,t)-hypergraph $\mathcal{H}$ contains multiple tower structures. That is, each edge in $\mathcal{H}$ is contained in $t-1$ towers which are disjoint except for that edge. Figure 3.9 shows a $(3,3)$-tower hypergraph and its Levi graph representation. Figure 3.10 shows a $(3,4)$-tower hypergraph.

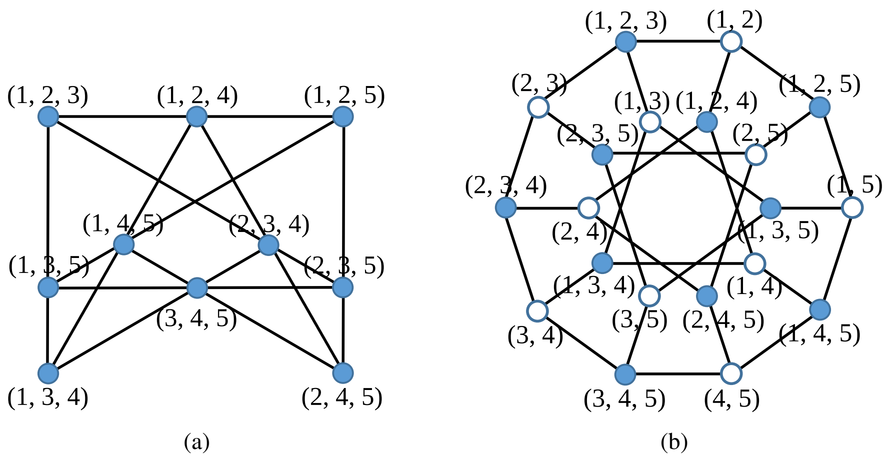


Figure 3.9: A $(3,3)$-tower hypergraph and its Levi graph representation. (a) A $(3,3)$-tower hypergraph; (b) A Levi graph representation of $(3,3)$-tower hypergraph.

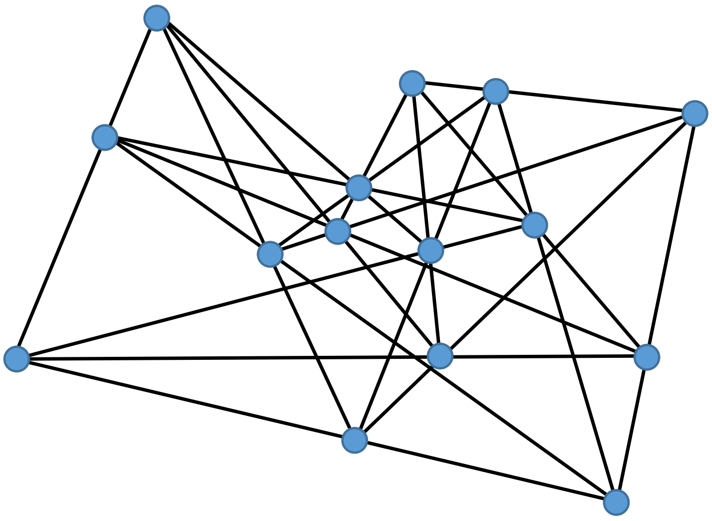


Figure 3.10: A $(3,4)$-tower hypergraph.

Definition 10 Consider an (r, t)-tower hypergraph $\mathcal{H}$. In $\mathcal{H}$, the vertices are associated with information symbols, and the edges are associated with parity symbols. Each parity symbol is calculated by binary addition of information symbols contained in the parity symbol. Then, we can convert $\mathcal{H}$ into a code $\mathcal{C}$. We call this code $\mathcal{C}$ an (r, t)-tower hypergraph code.

Consider a (r, t)-tower hypergraph $\mathcal{H}$ with v vertices and e edges. In $\mathcal{H}$, each edge contains r vertices, and each vertex is contained in t disjoint edges. Thus, $\mathcal{H}$ is also an (r, t)-hypergraph. Since $v = \binom{r+t-1}{t}$ and $e = \frac{vt}{r} = \binom{r+t-1}{r}$, the corresponding (r, t)-tower hypergraph code has length $\binom{r+t}{t}$, code rate $\frac{r}{r+t}$, and minimum distance $t + 1$.

Theorem 4 Let $\mathcal{C}$ be an (r, t)-tower hypergraph code. Then, $\mathcal{C}$ has $(r, t)_a$-availability.

Proof: Since $\mathcal{C}$ is also an (r, t)-hypergraph code, $\mathcal{C}$ has $(r, t)_i$-availability. Now, we will show every parity symbol also has (r, t)-availability. Consider a tower hypergraph $\mathcal{H}$ corresponding to $\mathcal{C}$ and an edge e_i in $\mathcal{H}$. For a $(t-2)$-subset of $t-1$ indices of e_i, the number of edges containing the $t-2$ indices is $r+1$, including e_i. We denote by F the edge set. Every vertex contained in any edge in F is shared by only two edges in F. Thus, all the edges in F form a dependent set, that is, e_i can be recovered by the other r edges in F. The number of such sets is $\binom{t-1}{t-2} = t - 1$, and all the sets share only e_i. Since e_i can be also recovered by r vertices contained in e_i, e_i has total t disjoint repair sets of size r. ■

From the proof of the Theorem 4, we can expect that, when a parity symbol is lost, a repair set is selected from the indices of vertices and edges in the corresponding tower hypergraph. For example, if an edge with $t - 1$ indices is lost, we can choose a set of edges which share $t-2$ indices among $t-1$ indices of the edge. We can also choose a set of all the vertices which contain $t-1$ indices of the edge.

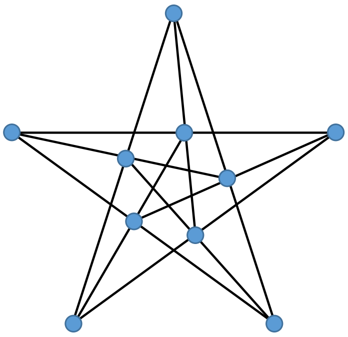


Figure 3.11: A $(3,3)$-hypergraph.

Remark 5 1. Figure 3.11 shows a $(3,3)$-hypergraph. We can not apply the labelling in Definition 9 to the hypergraph, even though it has $\binom{3+3-1}{3}=10$ vertices. Therefore, not every hypergraph is a tower hypergraph.

2. From the hypergraph in Figure 3.11, we can obtain a $[20,10,4]_2$ LRC with $(3,3)_i$-availability (but not $(3,3)_a$-availability) which is optimal. It can not be obtained from the construction in [30].

3.3 Extended Graph-based Optimal LRCs with Availability

We try to find some extensions which increase the minimum distance of hypergraph codes and make the extended codes also achieve the bound (3.2) with equality. Let $\mathcal{C}$ be an $[n,k,d]_2$ (r,t)-hypergraph code and G be the generator matrix of $\mathcal{C}$. We consider three cases: (1) d is odd. (2) k is odd and $r=2, t=k-1$. (3) $k \geqslant 8, 7 \nmid k$ and $r=t=3$. For the first case, we add a column of all ones to G. For the second case, we add p columns of all ones to G. Here, p is a positive integer less than or equal to $k-2$. For the third case, we add one column or two columns of all ones to G.

In the extension for case (1), since each row of G has odd weight, the minimum distance increases by one. To check the extension for case (2), recall the definition of $W(s)$ in Lemma 1. From the structure of G, $W(s) = s + s(k-s)$, and thus, $W(s)$ has the smallest value when $s = 1, k$ and the second smallest value when $s = 2, k-1$. Since $W(2) - W(1) = k-2$, the minimum distance increases by p, $1 \leqslant p \leqslant k-2$. Similarly, in the extension for case (3), from the structure of G, it is easy to see that $W(s)$ has the smallest value 4 when $s = 1$ and the second smallest value $\geqslant 6$ when $s = 2$. Thus, the minimum distance increases by one or two. For the case of $7|k$ and $r = t = 3$, the minimum distance does not increase since $W(s)$ has the smallest value when $s = 1, 4$.

Based on the extensions, we call the extended code with p columns of all ones a p-extended hypergraph code. If there are two extended hypergraph codes, $[n_1, k_1, d]_2$ p_1-extended (r, t)-hypergraph code and $[n_2, k_2, d]_2$ p_2-extended (r, t)-hypergraph code, using Remark 2-5), we can also obtain $[n_1+n_2, k_1+k_2, d]_2$ p-extended (r, t)-hypergraph codes for any positive integer p, $1 \leqslant p \leqslant \min(p_1, p_2)$.

We note that these extended (r, t)-hypergraph codes are optimal codes for the class 2) in Remark 1. All of the remaining cases which we do not consider in this paper are open. So, to find such codes would be an interesting research topic for optimal code design with new parameters. Our approach also provides insight into the relationship between the class 3) and 4) in Remark 1.

We confirmed that the proposed extensions work well by a computer simulation. Table 3.2 shows some examples of them.

Table 3.2: Hypergraph codes and their possible extensions

$[n,k,d]_2$ hypergraph codes	$(r,t)_i$-availability	Possible values for p
$[6,3,3]$	$(2,2)$	1
$[8,4,3]$	$(2,2)$	1
$[10,5,3]$	$(2,2)$	1
$[10,6,3]$	$(3,2)$	1
$[12,6,3]$	$(2,2)$	1
$[14,7,3]$	$(2,2)$	1
$[15,5,5]$	$(2,4)$	$1,2,3$
$[15,9,3]$	$(3,2)$	1
$[15,10,3]$	$(4,2)$	1
$[16,8,4]$	$(3,3)$	$1,2$
$[18,6,5]$	$(2,4)$	1
$[18,9,4]$	$(3,3)$	$1,2$
$[18,12,3]$	$(4,2)$	1
$[20,10,4]$	$(3,3)$	$1,2$
$[28,7,7]$	$(2,6)$	$1,2,3,4,5$
$[30,10,5]$	$(2,4)$	$1,2,3$

Chapter 4

Block-punctured Simplex Codes

In this chapter, we extend a subclass of graph-based LRCs into block-punctured Simplex codes named Combination codes [32]. Binary Simplex codes (dual of Hamming codes or punctured Hadamard codes) [1] are well-known LRCs attaining existing upper bounds [4, 17] on the code dimension and minimum distance which take into account the field size, locality, and availability. An $[n = 2^k - 1, k, d = 2^{k-1}]_2$ Simplex code has joint locality $(r_1, r_2)_a = (2, 3)$ [13] and $(2, d-1)_a$-availability [17]. Even though the Simplex codes have good joint locality and availability, they have extremely low code rate $\frac{k}{2^k-1}$. To obtain high rate codes without destroying the minimum distance, locality, and availability of the Simplex codes as possible, we propose a new construction of LRCs by puncturing the Simplex codes. We consider two types of the puncturing, “block-puncturing” and “subblock-puncturing”, as follows:

Block-puncturing: We consider a puncturing method which drops all the columns having weights larger than a given value in the generator matrix. We note that the authors of [33] also considered a similar puncturing method, called spherically puncturing, to increase the minimum distance of the first order Reed-Muller codes and Hadamard codes. They derived the minimum distance of two families of single-layer spherically punctured codes in closed-form. However, our block-punctured binary Simplex codes are a special case of multi-layer spherically punctured Hadamard codes, and thus, all the results of block-punctured binary Simplex codes are new.

Subblock-puncturing: We consider a fine puncturing method which drops few more columns of the largest weight after block-puncturing to increase the code rate more. To minimize the loss of the minimum distance, locality, and availability after the puncturing, we propose a heuristic algorithm. We confirmed that the algorithm works well by a computer simulation.

We determine the minimum distance, joint locality $(r_1, r_2)_a$, joint information locality, and availability of block-punctured binary Simplex codes in closed-form. Moreover, we define new generalized notions of availability, availability for multiple symbols, joint availability, and joint information availability, for LRCs. The joint availability presents how many disjoint repair sets there are for each erased symbol set of all the possible size. The joint information availability considers such property for only information symbols. We also determine the joint information availability of block-punctured binary Simplex codes in closed-form.

Recently, batch codes [34] and switch codes [35] which is a special case of batch codes with small query size are proposed in [36, 37], respectively. Here, the query size is the maximum number of codeword symbols used to recover an information symbol, and thus, it can be viewed analogous to information symbol locality of LRCs. Moreover, the minimum number of queries allowed for an information symbol can be viewed as the information symbol availability of LRCs. The concept of joint availability is new for batch codes as well as LRCs. Therefore, it can be applied to both areas.

Finally, we discuss the optimality of the proposed codes in terms of some well-known upper bounds on the code dimension and minimum distance which take into account the field size, locality, and availability. We also compare the code rates of several codes with various availabilities including the proposed ones. The result shows that the proposed ones have higher code rates than the currently known best codes under the condition of the same availability.

4.1 Block-punctured Simplex Codes

We start with the definition of the proposed codes, Combination (k, w) codes, as the following. In the definition, we denote $[z] \triangleq \{1, 2, \ldots, z\}$ for a positive integer $z \in \mathbb{Z}^+$.

Definition 11 Let two integers k and w satisfy $k \geqslant 3$ and $2 \leqslant w \leqslant k$. For $i \in [w]$, let G_i be the $k \times \binom{k}{i}$ matrix consisting of all the columns of weight i. Let $G = [G_1|G_2| \cdots |G_w]$. Then, the binary linear code $\mathcal{C}$ generated by G has length $\sum_{i=1}^{w} \binom{k}{i}$ and dimension k. We call $\mathcal{C}$ a Combination (k, w) code. □

In the remaining of this chapter, we will fix G the generator matrix defined in Definition 11 for Combination (k, w) codes. We note that the Combination $(k, 2)$ code and a Complete graph code are permutation equivalent, and that the Combination (k, k) code and a Simplex code [1] are permutation equivalent. We do not consider the case of $k \leqslant 2$ and the case of $w = 1$, since they are trivial. The Combination (k, w) code is the result of block-puncturing of the Simplex code. That is, G can be also obtained by deleting all the columns of weight larger than w from the generator matrix of the Simplex code.

$$G = \left(\begin{array}{c} \left.\begin{array}{c} 1\ 1\ \cdots\ 1\ 1\ 0\ 0\ \cdots\ 0\ 0 \\ \boxed{\qquad\qquad M' \qquad\qquad} \end{array}\right\} s \\ \\ \\ \end{array} \right)$$

Figure 4.1: The column-permuted version of G.

Later, we will further consider puncturing a few more columns of weight w.

Now, using the following Lemma 2 and Lemma 3, we derive the minimum distance of Combination (k, w) codes in Theorem 5.

Lemma 2 Let $\mathcal{C}$ be a Combination (k, w) code with its generator matrix G. For $s = 1, 2, \ldots, k$, a codeword obtained by adding any s rows of G has weight $W(s)$, which is given by the following:

$$W(s) = \sum_{i=1}^{w} \sum_{\substack{1 \leqslant j \leqslant i \\ j \text{ odd}}} \binom{s}{j} \binom{k-s}{i-j}. \tag{4.1}$$

Proof: For every $i \in [w]$, let M_i be a s-by-$\binom{k}{i}$ submatrix of G_i corresponding to the selected s rows. Then, $W(s)$ is the sum of the numbers of odd weight columns in M_i for all $i \in [w]$. Observe that the inner sum in (4.1) is the number of odd weight columns of M_i. ■

Lemma 3 Let $\mathcal{C}$ be a Combination (k, w) code with its generator matrix G. For $s = 2, 3, \ldots, k$, choose any s rows in G. We denote by $\mathbf{x}$ any one of them, and consider a submatrix M' consisting of the remaining $s - 1$ rows. We denote by $W_1(s-1)$ the number of odd weight columns in M' out of the columns corresponding to 1's positions of $\mathbf{x}$. We denote by $W_0(s-1)$, similarly, those corresponding to 0's positions of $\mathbf{x}$. Then, $W_1(s-1) \leqslant W_0(s-1)$.

Proof: Since each row of G is permutation equivalent, we can assume that any s rows are selected. It is also the same with $\mathbf{x}$. Then, $W_0(s-1)$ and $W_1(s-1)$ can be written as follows:

$$\begin{aligned} W_0(s-1) &= \sum_{i=1}^{w} \sum_{\substack{1 \leqslant j \leqslant i \\ j \text{ odd}}} \binom{s-1}{j} \binom{(k-1)-(s-1)}{i-j}, \\ W_1(s-1) &= \sum_{i=1}^{w} \sum_{\substack{1 \leqslant j \leqslant i-1 \\ j \text{ odd}}} \binom{s-1}{j} \binom{(k-1)-(s-1)}{(i-1)-j} \\ &= \sum_{i=1}^{w-1} \sum_{\substack{1 \leqslant j \leqslant i \\ j \text{ odd}}} \binom{s-1}{j} \binom{(k-1)-(s-1)}{i-j}. \end{aligned}$$

Therefore, we finally obtain $W_1(s-1) \leqslant W_0(s-1)$. ■

Theorem 5 Let $\mathcal{C}$ be a Combination (k, w) code. Then, the minimum distance d of $\mathcal{C}$ is

$$d = \sum_{i=1}^{w} \binom{k-1}{i-1}.$$

Proof: Without loss of generality, consider the top s rows in the generator matrix G of $\mathcal{C}$. We can always suppose that the first row of G is of the form $(1, \ldots, 1, 0, \ldots, 0)$ with column permutation as shown in Fig. 4.1. Then, $W(s)$ in Lemma 2 can be written as follows:

$$W(s) = \begin{cases} W(1), & \text{for } s = 1, \\ W(1) - W_1(s-1) + W_0(s-1), & \text{for } 2 \leqslant s \leqslant k, \end{cases}$$

where $W_1(s-1)$ and $W_0(s-1)$ are the numbers of odd weight columns defined in Lemma 3. Since $W_1(s-1) \leqslant W_0(s-1)$ by Lemma 3, we have $W(1) \leqslant W(s)$. Finally, using the result of Lemma 2, the minimum distance d is

$$d = \min_{s \in [k]} W(s) = W(1) = \sum_{i=1}^{w} \binom{k-1}{i-1}.$$

■

Remark 6 For two integers $k \geqslant 3$ and $2 \leqslant w \leqslant k$, we specially denote the minimum distance of a Combination (k, w) code by $d(k, w)$ instead of d. Then, we have $d(k, 2) = k \leqslant d(k, 3) = \frac{1}{2}(k^2 - k + 2) \leqslant \cdots \leqslant d(k, k) = 2^{k-1}$ and $d(k, w) \geqslant 2^{w-1}$ from Theorem 5.

Theorem 6 Let $\mathcal{C}$ be a Combination (k, w) code. Then, $\mathcal{C}$ has joint locality $(r_1, r_2)_a = (2, 3)$.

Proof: Consider the generator matrix G of $\mathcal{C}$. For $r_1 = 2$, we have to show that an erased column of G can be expressed as a linear combination of at most two other columns. For $r_2 = 3$, we have to show that two erased columns can be expressed as a linear combination of at most three other columns. More precisely, consider the following two cases.

1. For one erasure, let $\mathbf{g}_{E_1}$ be the erased column and E_1 be the set of non-zero row indices of $\mathbf{g}_{E_1}$. Recall that G contains every column of weight $1, 2, \ldots, |E_1|$. Then, there exists a column $\mathbf{g}_{R_1}$ as shown in Fig. 4.2. Then, the third column $\mathbf{g}_{R_2} = \mathbf{g}_{E_1} + \mathbf{g}_{R_1}$ exists in G.
2. For two erasures, let $\mathbf{g}_{E_1}$ and $\mathbf{g}_{E_2}$ be the two erased columns, and E_1 and E_2 be the sets of non-zero row indices of $\mathbf{g}_{E_1}$ and $\mathbf{g}_{E_2}$, respectively. Without loss

of generality, assume that $|E_2| \leqslant |E_1|$. Recall that G contains every column of weight $1, 2, \ldots, |E_1|$. Then, there exists a column $\mathbf{g}_{R_1}$ as shown in Fig. 4.3. Then two other columns $\mathbf{g}_{R_2} = \mathbf{g}_{E_1} + \mathbf{g}_{R_1}$ and $\mathbf{g}_{R_3} = \mathbf{g}_{E_2} + \mathbf{g}_{R_1}$ can be found in G.

■

Remark 7 For three integers $l \geqslant 3$, $k \geqslant 3$, and $2 \leqslant w \leqslant k$, we specially denote the l-locality of a Combination (k, w) code by $r_l(k, w)$ instead of r_l. Then, we have $r_l(k, k) \leqslant r_l(k, k-1) \leqslant \cdots \leqslant r_l(k, 2)$. In [11], the authors showed that, for an $[n = 2^k - 1, k, d = 2^{k-1}]_2$ simplex code, any $l \leqslant d-1$ erased code symbols can be corrected by contacting at most $l+1$ other code symbols. From the result, we can expect the range of the value $r_l(k, w)$. Moreover, from the structure of generator matrix, we can easily check that $(r_1(k,k), r_2(k,k), r_3(k,k)) = (2,3,3)$ for $k \geqslant 3$, $(r_1(k,2), r_2(k,2), r_3(k,2)) = (2,3,4)$ for $k = 4$, and $(r_1(k,2), r_2(k,2), r_3(k,2)) = (2,3,5)$ for $k \geqslant 5$. These are more useful to obtain exact lower and upper bounds of the value $r_l(k, w)$ for $l = 1, 2, 3$.

As mentioned in Remark 7, even though a bound of l-locality for the Simplex codes is introduced in [11], it is not easy to find the exact value of l-locality for Combination (k, w) codes. In this paper, as a first step, we provide the joint information locality of Combination (k, w) codes as the following.

Theorem 7 Let $\mathcal{C}$ be a Combination (k, w) code. Then, $\mathcal{C}$ has joint information locality

$$r_l = \begin{cases} l, & \text{for } l, w \geqslant 3, \\ l+1, & \text{otherwise.} \end{cases}$$

Proof: For $l = 1$ and 2, we have $r_l = l + 1$ from Fig. 4.2-(a) and Fig. 4.3-(a) in Theorem 6. For $l \geqslant 3$, we consider two cases: (1) $w = 2$. (2) $w \geqslant 3$.

Consider the generator matrix G of $\mathcal{C}$. Let $\mathbf{g}_{E_1}, \mathbf{g}_{E_2}, \ldots, \mathbf{g}_{E_l}$ be the erased columns.

Case (1): For $l \geqslant 3$, $w = 2$, the minimum distance d is k, and thus $l \leqslant k-1$. Then, we can always choose a column $\mathbf{g}_A$ of weight 1 which is not erased. Using the column, we can reconstruct the erased columns from the following $l+1$ other columns.

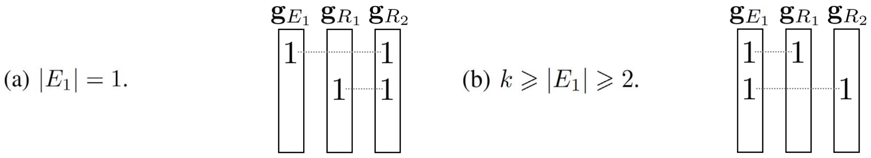
$$\begin{aligned}
\mathbf{g}_{R_1} &= \mathbf{g}_A, \\
\mathbf{g}_{R_2} &= \mathbf{g}_A + \mathbf{g}_{E_1}, \\
\mathbf{g}_{R_3} &= \mathbf{g}_A + \mathbf{g}_{E_2}, \\
&\vdots \\
\mathbf{g}_{R_{l+1}} &= \mathbf{g}_A + \mathbf{g}_l.
\end{aligned}$$

Case (2): For $l, w \geqslant 3$, we can always reconstruct the erased columns from the following l other columns.

$$\begin{aligned}
\mathbf{g}_{R_1} &= \mathbf{g}_{E_1} + \mathbf{g}_{E_2} + \mathbf{g}_{E_3}, \\
\mathbf{g}_{R_2} &= \mathbf{g}_{E_1} + \mathbf{g}_{E_2}, \\
\mathbf{g}_{R_3} &= \mathbf{g}_{E_1} + \mathbf{g}_{E_3}, \\
&\vdots \\
\mathbf{g}_{R_l} &= \mathbf{g}_{E_1} + \mathbf{g}_{E_l}.
\end{aligned}$$

From the above, we obtain the joint information locality of $\mathcal{C}$. ■

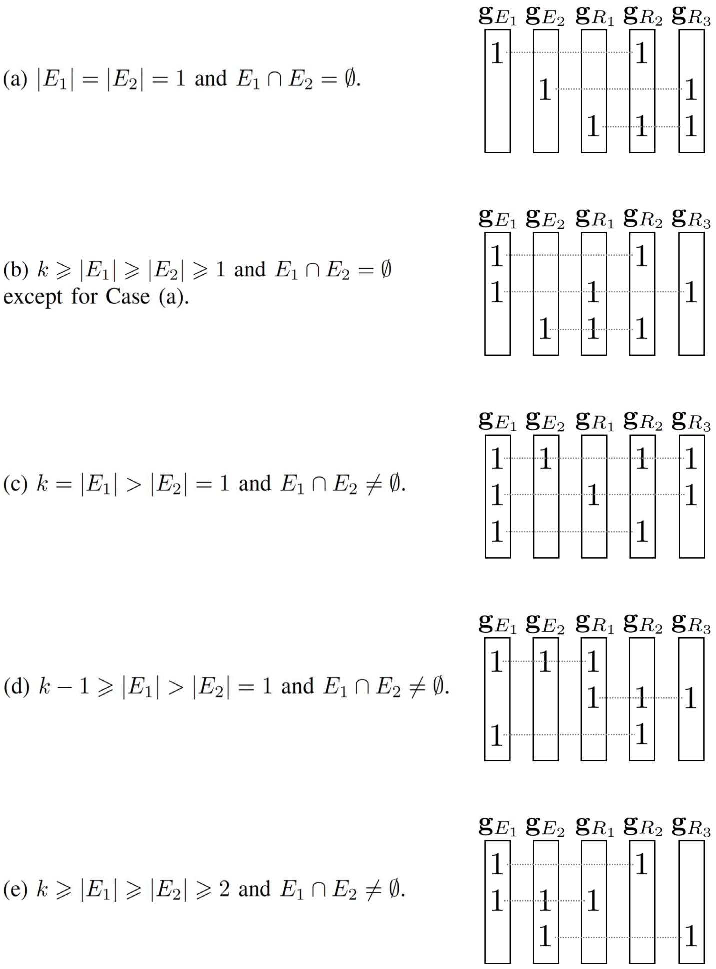
Now, using the following Lemma 4 and Lemma 5, we derive the availability of Combination (k, w) codes in Theorem 8.


$\mathbf{g}_{E_1}$ $\mathbf{g}_{R_1}$ $\mathbf{g}_{R_2}$
$\mathbf{g}_{E_1}$ $\mathbf{g}_{R_1}$ $\mathbf{g}_{R_2}$

(a) $|E_1| = 1$.

(b) $k \geqslant |E_1| \geqslant 2$.

Figure 4.2: Two cases for $r_1 = 2$.


$\mathbf{g}_{E_1}$ $\mathbf{g}_{E_2}$ $\mathbf{g}_{R_1}$ $\mathbf{g}_{R_2}$ $\mathbf{g}_{R_3}$

(a) $|E_1| = |E_2| = 1$ and $E_1 \cap E_2 = \emptyset$.

(b) $k \geqslant |E_1| \geqslant |E_2| \geqslant 1$ and $E_1 \cap E_2 = \emptyset$ except for Case (a).

(c) $k = |E_1| > |E_2| = 1$ and $E_1 \cap E_2 \neq \emptyset$.

(d) $k - 1 \geqslant |E_1| > |E_2| = 1$ and $E_1 \cap E_2 \neq \emptyset$.

(e) $k \geqslant |E_1| \geqslant |E_2| \geqslant 2$ and $E_1 \cap E_2 \neq \emptyset$.

Figure 4.3: Five cases for $r_2 = 3$.

Lemma 4 Let $\mathcal{C}$ be a Combination (k, w) code with its generator matrix G. Then, the availability $T(i)$ of a symbol corresponding to a column of weight i in G is

$$T(i) = \frac{1}{2} \sum_{u=0}^{i} \binom{i}{u} \sum_{v=0}^{\min(w-u, w-(i-u))} \binom{k-i}{v} - 1. \tag{4.2}$$

Proof: Recall that 1-locality r_1 is 2 for Combination codes by Theorem 6. The availability of a symbol is, therefore, the number of disjoint sets of two columns to repair the corresponding column in G. More precisely, assume that A is the set of non-zero row indices for an erased column of weight i. Then, the availability of the symbol corresponding to A is the number of disjoint choices for two distinct sets of integers, B and C, such that $A = (B \cup C) - (B \cap C)$ where $|A| = i$, and $0 < |C| \leqslant |B| \leqslant w$. There are three cases for the choice:

1. If $B \cap C = C$, the number of choices is

$$\sum_{v=1}^{w-i} \binom{k-i}{v}.$$

2. If $B \cap C = \varnothing$, the number of choices is

$$\frac{1}{2} \sum_{u=1}^{i-1} \binom{i}{u}.$$

3. Otherwise, the number of choices is

$$\frac{1}{2} \sum_{u=1}^{i-1} \binom{i}{u} \sum_{v=1}^{\min(w-u, w-(i-u))} \binom{k-i}{v}.$$

The sum of the above three gives (4.2). ■

Lemma 5 Let $\mathcal{C}$ be a Combination (k, w) code with its generator matrix G. For a possible positive integers i, let $T(i)$ be the availability of a symbol corresponding to the column of weight i in G. Then, $T(\cdot)$ satisfies the following:

$$T(2j-1) = T(2j) \geqslant T(2j+1), \tag{4.3}$$

i.e. $T(1) = T(2) \geqslant T(3) = T(4) \geqslant T(5) = T(6) \cdots T(w)$.

Proof: Observe that there are $\binom{k}{i}$ columns of weight i in G, and all the $\binom{k}{i}$ symbols corresponding to these columns have the same availability, which is denoted by $T(i)$. Now, we rewrite (4.2) as follows

$$\begin{aligned} T(2j-1) &= \sum_{u=j}^{2j-1} \binom{2j-1}{u} \sum_{v=0}^{w-u} \binom{k-(2j-1)}{v} - 1, \\ T(2j) &= \sum_{u=j}^{2j} \binom{2j}{u} \sum_{v=0}^{w-u} \binom{k-2j}{v} - \frac{1}{2}\binom{2j}{j} \sum_{v=0}^{w-j} \binom{k-2j}{v} - 1. \end{aligned}$$

Now, we show that $T(2j-1) - T(2j) = 0$.

$$\begin{aligned} &T(2j-1) - T(2j) \\ &= \sum_{u=j}^{2j-1} \binom{2j-1}{u} \sum_{v=0}^{w-u} \left[\binom{k-2j}{v} + \binom{k-2j}{v-1} \right] \\ &\quad - \left[\sum_{u=j}^{2j-1} \left[\binom{2j-1}{u} + \binom{2j-1}{u-1} \right] \sum_{v=0}^{w-u} \binom{k-2j}{v} + \sum_{v=0}^{w-2j} \binom{k-2j}{v} \right] \\ &\quad + \binom{2j-1}{j-1} \sum_{v=0}^{w-j} \binom{k-2j}{v} \\ &= \sum_{u=j}^{2j-1} \left[\binom{2j-1}{u} \sum_{v=0}^{w-u} \binom{k-2j}{v-1} - \binom{2j-1}{u-1} \sum_{v=0}^{w-u} \binom{k-2j}{v} \right] \\ &\quad - \sum_{v=0}^{w-2j} \binom{k-2j}{v} + \binom{2j-1}{j-1} \sum_{v=0}^{w-j} \binom{k-2j}{v} \end{aligned}$$

$$
\begin{aligned}
&= \left[\sum_{v=0}^{w-2j}\binom{k-2j}{v} - \binom{2j-1}{j-1}\sum_{v=0}^{w-j}\binom{k-2j}{v}\right] \\
&\quad - \sum_{v=0}^{w-2j}\binom{k-2j}{v} + \binom{2j-1}{j-1}\sum_{v=0}^{w-j}\binom{k-2j}{v} = 0.
\end{aligned}
$$

This gives $T(2j-1) = T(2j)$.

For the inequality in (4.3), we rewrite (4.2) as follows

$$
\begin{aligned}
T(2j) &= \sum_{u=j+1}^{2j}\binom{2j}{u}\sum_{v=0}^{w-u}\binom{k-2j}{v} + \frac{1}{2}\binom{2j}{j}\sum_{v=0}^{w-j}\binom{k-2j}{v} - 1, \\
T(2j+1) &= \sum_{u=j+1}^{2j+1}\binom{2j+1}{u}\sum_{v=0}^{w-u}\binom{k-(2j+1)}{v} - 1.
\end{aligned}
$$

Now, it is easy to check that $T(2j) - T(2j+1) \geqslant 0$.

$$
\begin{aligned}
&T(2j) - T(2j+1) \\
&= \sum_{u=j+1}^{2j}\binom{2j}{u}\sum_{v=0}^{w-u}\left[\binom{k-(2j+1)}{v} + \binom{k-(2j+1)}{v-1}\right] + \binom{2j-1}{j-1}\sum_{v=0}^{w-j}\binom{k-2j}{v} \\
&\quad - \sum_{u=j+1}^{2j}\left[\binom{2j}{u} + \binom{2j}{u-1}\right]\sum_{v=0}^{w-u}\binom{k-(2j+1)}{v} - \sum_{v=0}^{w-(2j+1)}\binom{k-(2j+1)}{v} \\
&= \sum_{u=j+1}^{2j}\left[\binom{2j}{u}\sum_{v=0}^{w-u}\binom{k-(2j+1)}{v-1} - \binom{2j}{u-1}\sum_{v=0}^{w-u}\binom{k-(2j+1)}{v}\right] \\
&\quad + \binom{2j-1}{j-1}\sum_{v=0}^{w-j}\binom{k-2j}{v} - \sum_{v=0}^{w-(2j+1)}\binom{k-(2j+1)}{v} \\
&= \left[\sum_{v=0}^{w-(2j+1)}\binom{k-(2j+1)}{v} - \binom{2j}{j}\sum_{v=0}^{w-(j+1)}\binom{k-(2j+1)}{v}\right] \\
&\quad + \binom{2j-1}{j-1}\sum_{v=0}^{w-j}\binom{k-2j}{v} - \sum_{v=0}^{w-(2j+1)}\binom{k-(2j+1)}{v}
\end{aligned}
$$

$$= \binom{2j-1}{j-1}\left[\sum_{v=0}^{w-j}\left[\binom{k-(2j+1)}{v}+\binom{k-(2j+1)}{v-1}\right]-2\sum_{v=0}^{w-(j+1)}\binom{k-(2j+1)}{v}\right]$$
$$= \binom{2j-1}{j-1}\binom{k-(2j+1)}{w-j} \geqslant 0.$$

■

Theorem 8 Let $\mathcal{C}$ be a Combination (k, w) code. Then, the availability t of $\mathcal{C}$ is

$$t = \frac{1}{2}\sum_{u=0}^{w}\binom{w}{u}\sum_{v=0}^{\min(w-u,u)}\binom{k-w}{v} - 1.$$

In particular, $t = d - 1$ when $w = 2, k-1$ or k, and $t = d - 2$ when $w = k - 2 > 2$.

Proof: By Lemma 4 and Lemma 5, the availability t of $\mathcal{C}$ becomes

$$t = \min_{i\in[w]} T(i) = T(w) = \frac{1}{2}\sum_{u=0}^{w}\binom{w}{u}\sum_{v=0}^{\min(w-u,u)}\binom{k-w}{v} - 1.$$

■

We note that we can obtain the availabilities of a Complete graph code and a binary Simplex code [1] using a Combination $(k, 2)$ code and a Combination (k, k) code, respectively. The availability of a Complete graph code is also obtained from its graph representation, and that of a binary Simplex code is also obtained from its one-step majority-logic decoding structure [17, 38].

Remark 8 For two integers $k \geqslant 3$ and $2 \leqslant w \leqslant k$, we specially denote the availability of a Combination (k, w) code by $t(k, w)$ instead of t. Then, we have $t(k, 2) = k - 1 \leqslant t(k, 3) = 3k - 6 \leqslant \cdots \leqslant t(k, k) = 2^{k-1} - 1$ and $t(k, w) \geqslant 2^{w-1} - 1$ from Theorem 8.

Example 3 Table 4.1 shows the values of parameters of Combination (k, w) codes for $k = 8$ and $2 \leqslant w \leqslant k$. All the values of minimum distances and availabilities for the codes are calculated by using Theorem 5 and Theorem 8.

Table 4.1: Parameters of Combination $(k = 8, w)$ codes with joint locality $(r_1, r_2)_a = (2, 3)$

Maximum column weight w	Code length n	Code rate k/n	Mininum distance d	Availability t
2	36	$2/9$	8	7
3	92	$4/41$	29	18
4	162	$4/81$	64	53
5	218	$4/109$	99	90
6	246	$4/123$	120	118
7	254	$4/127$	127	126
8	255	$8/255$	128	127

To increase the code rate more, we propose another puncturing method, called subblock-puncturing, which deletes a few more columns of weight w from the generator matrix of a Combination (k, w) code. Let p be the number of punctured columns, called puncturing length. If p becomes $\binom{k}{w}$, the punctured code is the same with a Combination $(k, w-1)$ code. Thus, we only consider $1 \leqslant p \leqslant \binom{k}{w} - 1$.

Example 4 Let $\mathcal{C}$ be a Combination $(4, 2)$ code. Then, its generator matrix is

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

Consider a case of the puncturing length $p = 2$. If we delete the last two columns, the minimum distance of the punctured code becomes 2. However, if we delete column 5 and 10, the minimum distance of the punctured code becomes 3. From this observation, we propose a simple heuristic method for the subblock-puncturing as the following.

Consider the generator matrix $G = [G_1|G_2|\cdots|G_w]$ of a Combination (k, w) code. To obtain a punctured Combination (k, w) code by the subblock-puncturing, we use the following generator matrix construction algorithm.

Algorithm 1 Construction of the generator matrix of a punctured Combination (k, w) code by the subblock-puncturing

```
Set $x_{min} = \binom{k}{w}$.
for every column set of size $p$ in $G_w$ do
    Make a $k \times p$ matrix $\Phi$.
    Calculate the difference $x$ between maximum and minimum weights of rows in $\Phi$.
    if $x_{min} \geqslant x$ then
        Set $x_{min} = x$ and $\Phi_{min} = \Phi$.
    end if
end for
Delete columns of $G$ which are the same with those of $\Phi_{min}$.
```

Using the subblock-puncturing, we can obtain optimal codes in terms of the following bounds. For LRCs with $(r_1, t)_i$-availability, an upper bound on the minimum distance was presented in [15] under the following condition. Let $\mathcal{C}$ be an $[n, k, d]_q$ LRC. Every information symbol of $\mathcal{C}$ has t disjoint repair sets, each set of size at most r_1, such that any repair set contains only one parity symbol. Then, the minimum distance d of $\mathcal{C}$ is bounded by

$$d \leqslant n - k - \left\lceil \frac{kt}{r_1} \right\rceil + t + 1. \tag{4.4}$$

Additionally, we note that the bound (4.4) implies

$$\frac{k}{n} \leqslant \frac{r_1}{r_1 + t}. \tag{4.5}$$

Now, consider a Combination $(k, 2)$ code and puncture the code using Algorithm 1. For a positive integer τ such that $2|k\tau$, when the puncturing length $p = \frac{k}{2}(k - \tau - 1)$, we obtain an $[n = k + \frac{k\tau}{2}, k, d]_2$ punctured Combination $(k, 2)$ code. Let $\mathcal{C}$ be the punctured code. It is easy to see that $\mathcal{C}$ has the $(2, \tau)_a$-availability and minimum distance $d = \tau + 1$, and thus, $\mathcal{C}$ is optimal in terms of the bounds (4.4) and (4.5). In particular, for $\tau = k - 1$, the punctured code becomes a Complete graph code and, for $\tau = k - \frac{k}{\gamma}$ where γ is a positive integer such that $2 \leqslant \gamma \leqslant k$ and $\gamma | k$, it becomes a Complete multipartite graph code. Moreover, for $\tau = 1$, we have optimal binary LRCs with the code rate $2/3$ and, for $\tau = 2$, we have optimal binary LRCs with the code rate $1/2$.

4.2 Optimality of Block-punctured Simplex Codes

In [39], for $[n, k, d]_q$ code with $(r_1, t)_a$-availability, an upper bound on the code rate is given by

$$\frac{k}{n} \leqslant \frac{1}{\prod_{j=1}^{t}(1 + \frac{1}{j \cdot r_1})}. \tag{4.6}$$

Unfortunately, for $t \geqslant 2$, it is not known whether the bound (4.6) is achievable. A construction of binary linear codes achieving any given (r_1, t)-availability was introduced in [30]. To the best of our knowledge, in terms of the code rate, this is the best known construction of codes achieving arbitrary (r_1, t)-availability. Since $r_1 \geqslant 2$ except for the repetition codes, we are more interested in $(2, t)$-availability case. There are also some existing codes achieving this availability: the codes from [30], Direct product codes [39, 40], and Simplex codes [1] (only for those values of $t = 2^{k-1} - 1$).

In Figure 4.4, we compare the code rates of the above mentioned codes and the proposed codes, both Combination codes and punctured Combination codes. Figure 4.4 shows the code rates of various codes achieving $(2, t)$-availability versus the value t from 1 to 15, together with the bound (4.6). From the figure, we confirm that the proposed codes have higher code rates than the codes from [30] (best known codes) as well as Direct product codes [39, 40] when the same $(2, t)$-availability is maintained.

In Figure 4.5 and Figure 4.6, we also compare the minimum distances and code lengths of the three families of the codes, the proposed codes, Simplex codes, and codes in [30], respectively. From these results, we confirmed that the proposed codes are attractive not only for code rate but also for other code parameters.

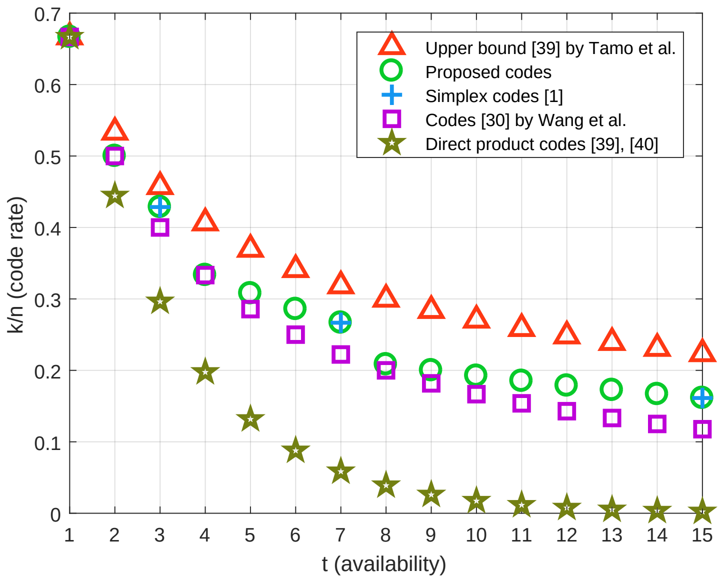


Figure 4.4: Comparison of the code rates for $r_1 = 2, 1 \leqslant t \leqslant 15$: For $t \geqslant 2$, all the proposed codes have joint locality $(r_1, r_2)_a = (2, 3)$.

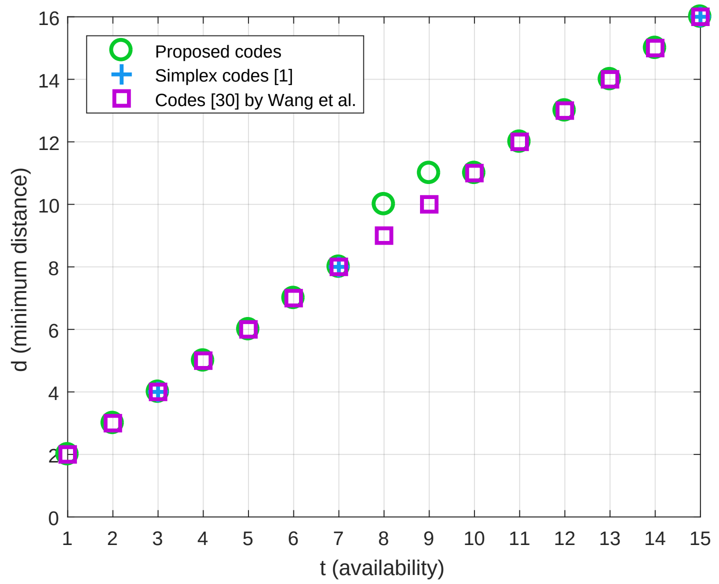


Figure 4.5: Comparison of the minimum distances for $r_1 = 2, 1 \leqslant t \leqslant 15$: For $t \geqslant 2$, all the proposed codes have joint locality $(r_1, r_2)_a = (2, 3)$.

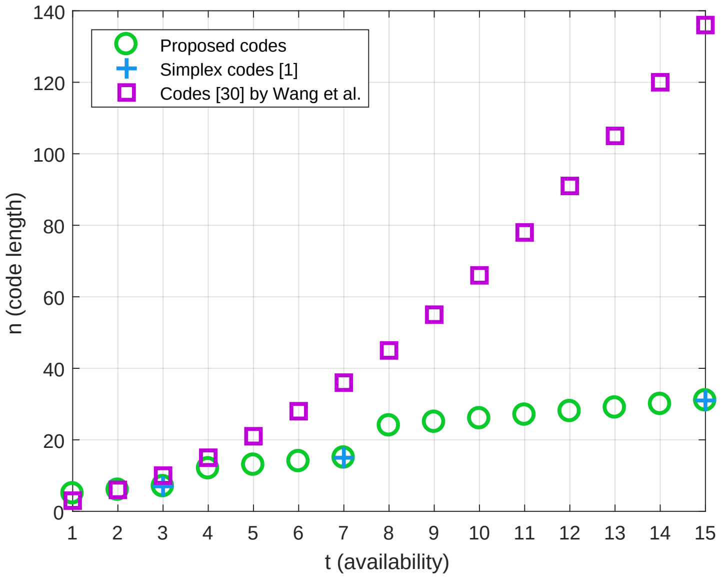


Figure 4.6: Comparison of the code lengths for $r_1 = 2, 1 \leqslant t \leqslant 15$: For $t \geqslant 2$, all the proposed codes have joint locality $(r_1, r_2)_a = (2, 3)$.

Chapter 5

New Bounds for Locally Repairable Codes

In this chapter, we suggest new alphabet-dependent upper bounds on the code dimension and minimum distance for LRCs with joint information availability [41]. From the bounds, we derive some well-known alphabet-dependent bounds and Singleton-like bounds without the alphabet constraint.

5.1 Joint Availability for LRCs

We first introduce some generalized notions of locality and availability for LRCs. We suggest joint locality which consider r_l's for multiple values of l instead of a single value of l. For example, when we consider $l = 1$ and $l = 2$ together, the joint locality is represented by $(r_1, r_2)_a$. If a code is systematic and the property applies only to its information symbols, then the code has joint information locality. Second, we consider the availability for multiple symbols. For example, if the central unit is interested in recovering 2 symbols, then to fetch it in parallel, it would need disjoint sets of at most r_2 other symbols to fetch them simultaneously. It is represented by (r_2, t_2)-availability.

Similar to joint locality, we also define joint availability and joint information availability for any decodable set of erased symbols. The detailed definitions are as follows.

Definition 12 Let $\mathcal{C}$ be an LRC. A set of localities $\{r_l | l \in [d-1]\}$ is called the joint locality of $\mathcal{C}$ if every symbol set of cardinality l can be recovered by at most r_l other symbols. □

Definition 13 Let $\mathcal{C}$ be an LRC. A set of localities $\{r_l | l \in [d-1]\}$ is called the joint information locality of $\mathcal{C}$ if every information symbol set of cardinality l can be recovered by at most r_l other symbols. □

Definition 14 Let $\mathcal{C}$ be an $(n, k, d)_q$ code. For a positive integer $l \in [d-1]$, a symbol set $\mathcal{E}$ of $\mathcal{C}$, $|\mathcal{E}| = l$, is said to have availability (r_l, t_l) if the symbol set $\mathcal{E}$ can be recovered from any single set of t_l disjoint symbol sets indexed by $\mathcal{R}_1(\mathcal{E}), \mathcal{R}_2(\mathcal{E}), \ldots, \mathcal{R}_{t_l}(\mathcal{E}) \subseteq [n] \setminus \mathcal{E}$ such that $|\mathcal{R}_j(\mathcal{E})| \leqslant r_l$, for all $j \in [t_l]$. □

Definition 15 Let $\mathcal{C}$ be an $(n, k, d)_q$ code. A set of availabilities $\{(r_l, t_l) : l \in [d-1]\}$ is called joint availability of $\mathcal{C}$ if every symbol set of cardinality l of $\mathcal{C}$ has the availability (r_l, t_l). □

Definition 16 Let $\mathcal{C}$ be a systematic $(n, k, d)_q$ code. A set of availabilities $\{(r_l, t_l) : l \in [d-1]\}$ is called joint information availability of $\mathcal{C}$ if every information symbol set of cardinality l of $\mathcal{C}$ has the availability (r_l, t_l). □

Now, we derive the joint information availability of Combination (k, w) codes from Theorem 7.

Theorem 9 Let $\mathcal{C}$ be a Combination (k, w) code. Then, $\mathcal{C}$ has joint information availability

$$(r_l, t_l)_i = \begin{cases} (l+1, \sum_{v=1}^{\min(w-1,k-l)} \binom{k-l}{v}), & \text{for } l = 1, 2, \\ (l+1, k-l), & \text{for } l \geqslant 3, w = 2, \\ (l, \min(l, w) - 2), & \text{otherwise.} \end{cases}$$

Proof: By Theorem 7, $\mathcal{C}$ has joint information locality $(r_1, r_2)_i = (2, 3)$. Consider the generator matrix G of $\mathcal{C}$. To obtain the value of t_1, let $\mathbf{g}_{E_1}$ be the erased column. Choose a column $\mathbf{g}_A$ of weight $a < w$, $A \neq E_1$. Then, we can reconstruct the erased column from the following two other columns.

$$\mathbf{g}_{R_1} = \mathbf{g}_A, \qquad \mathbf{g}_{R_2} = \mathbf{g}_A + \mathbf{g}_{E_1}.$$

When a is fixed, $\binom{k-1}{a}$ disjoint repairs sets exist, and there are $w - 1$ choices for a. Therefore, $t_1 = \sum_{v=1}^{w-1} \binom{k-1}{v}$.

Now, to obtain the value of t_2, let $\mathbf{g}_{E_1}$ and $\mathbf{g}_{E_2}$ be the erased columns. Choose a column $\mathbf{g}_A$ of weight $a < w$, $A \neq E_1, E_2$. Then, we can reconstruct the erased columns from the following three other columns.

$$\mathbf{g}_{R_1} = \mathbf{g}_A, \qquad \mathbf{g}_{R_2} = \mathbf{g}_A + \mathbf{g}_{E_1}, \qquad \mathbf{g}_{R_3} = \mathbf{g}_A + \mathbf{g}_{E_2}.$$

When a is fixed, $\binom{k-2}{a}$ disjoint repairs sets exist. When $w < k$, there are $w-1$ choices for a and, when $w = k$, there are $k-2$ choices for a. Therefore, $t_2 = \sum_{v=1}^{\min(w-1,k-2)} \binom{k-2}{v}$.

Now, consider the case of $l \geqslant 3$. Since $w \geqslant 2$ by the definition, we consider two cases: (1) $l \geqslant 3$, $w = 2$. (2) $l, w \geqslant 3$. Consider the generator matrix G of $\mathcal{C}$. Let $\mathbf{g}_{E_1}, \mathbf{g}_{E_2}, \ldots, \mathbf{g}_{E_l}$ be the erased columns.

Case (1): For $l \geqslant 3$, $w = 2$, the minimum distance d is k, and thus $l \leqslant k - 1$. Then, we can always choose a column $\mathbf{g}_A$ of weight 1 which is not erased. Using the column, we can reconstruct the erased columns from the following $l+1$ other columns.

$$\begin{aligned}
\mathbf{g}_{R_1} &= \mathbf{g}_A, \\
\mathbf{g}_{R_2} &= \mathbf{g}_A + \mathbf{g}_{E_1}, \\
\mathbf{g}_{R_3} &= \mathbf{g}_A + \mathbf{g}_{E_2}, \\
&\vdots \\
\mathbf{g}_{R_{l+1}} &= \mathbf{g}_A + \mathbf{g}_l.
\end{aligned}$$

For a fixed column $\mathbf{g}_A$, there is only one disjoint repair set. The number of choices of such a column $\mathbf{g}_A$ is $k-l$, and the repair sets are disjoint each other.

Case (2): For $l, w \geqslant 3$, we can always reconstruct the erased columns from the following l other columns. Here, i is a positive integer, $3 \leqslant i \leqslant \min(l, w)$.

$$\begin{aligned}
\mathbf{g}_{R_1} &= \sum_{u=1}^{i} \mathbf{g}_{E_u}, \\
\mathbf{g}_{R_2} &= \sum_{\substack{1 \leqslant u \leqslant i \\ u \neq 1}} \mathbf{g}_{E_u}, \\
&\vdots \\
\mathbf{g}_{R_i} &= \sum_{\substack{1 \leqslant u \leqslant i \\ u \neq i-1}} \mathbf{g}_{E_u}, \\
\mathbf{g}_{R_{i+1}} &= \sum_{u=1}^{i-2} \mathbf{g}_{E_u} + \mathbf{g}_{E_{i+1}}, \\
&\vdots \\
\mathbf{g}_{R_l} &= \sum_{u=1}^{i-2} \mathbf{g}_{E_u} + \mathbf{g}_{E_l}.
\end{aligned}$$

For a fixed value i, there is only one disjoint repair set. For all the possible i, the repair sets are disjoint each other.

From the above, we finally obtain the joint information availability of $\mathcal{C}$. ■

5.2 Alphabet-dependent Bounds for LRCs with Joint Information Availability

We propose new alphabet-dependent bounds for LRCs with joint information availability. We note that, since an LRC with joint availability is also an LRC with joint information availability, the proposed bounds also hold for LRCs with joint availability.

Theorem 10 For an $(n, k, d)_q$ code $\mathcal{C}$ that has joint information availability $\{(r_l, t_l) : l \in [d-1]\}$, we have

$$k \leqslant \min_{\substack{z \in \mathbb{Z}^+ \\ \boldsymbol{l}=\{l_j \in [d-1], 1 \leqslant j \leqslant z\} \\ \boldsymbol{y}=\{y_j \in [t_{l_j}], 1 \leqslant j \leqslant z\} \\ A(\boldsymbol{l},\boldsymbol{y})<k}} \left[A(\boldsymbol{l},\boldsymbol{y}) + k_{opt}^{(q)}\left(n - B(\boldsymbol{l},\boldsymbol{y}),\ d\right)\right], \tag{5.1}$$

where

$$A(\boldsymbol{l},\boldsymbol{y}) = \sum_{j=1}^{z}((r_{l_j} - 1)y_j + 1), \quad B(\boldsymbol{l},\boldsymbol{y}) = \sum_{j=1}^{z}(r_{l_j} y_j + l_j),$$

and $k_{opt}^{(q)}(n', d')$ is the largest possible dimension of a q-ary code of length n' and minimum distance d'.

Proof: We follow similar steps to the proofs in [4, 14]. We define $H(\mathbf{Y}_{\mathcal{I}}) = \log_q |\{\mathbf{Y}_{\mathcal{I}} : \mathbf{Y} \in \mathcal{C}\}|$ where $\mathbf{Y}_{\mathcal{I}}$ is the restriction of $\mathbf{Y}$ to a subset of coordinates $\mathcal{I} \subset [n]$. First of all, we show that, for any positive integer z satisfying $A(\boldsymbol{l},\boldsymbol{y}) < k$, there exists a set $\mathcal{I}$ such that $|\mathcal{I}| = B(\boldsymbol{l},\boldsymbol{y})$ and $H(\mathbf{Y}_{\mathcal{I}}) \leqslant A(\boldsymbol{l},\boldsymbol{y})$. To prove this, we construct the set $\mathcal{I}$ using the following algorithm. In the algorithm, without loss of generality, we assume that the first k symbols of the code $\mathcal{C}$ form an information symbol set.

Algorithm 2 Construction of $\mathcal{I}$

```
1:  Let $j = 1$ and $\mathcal{I}_0 = \emptyset$.
2:  while $H(\mathbf{Y}_{\mathcal{I}_{j-1}}) < k$ do
3:      Pick $\mathcal{E}_j \subset [k] \setminus \mathcal{I}_{j-1}$, $|\mathcal{E}_j| = l_j$ such that
        $H(\mathbf{Y}_{\mathcal{I}_{j-1}}) < H(\mathbf{Y}_{\mathcal{I}_{j-1} \cup \mathcal{E}_j})$.
4:      Pick $y_j \leqslant t_{l_j}$ disjoint repair sets of $\mathcal{E}_j$, $\mathcal{R}_1(\mathcal{E}_j)$,
        $\mathcal{R}_2(\mathcal{E}_j), \ldots, \mathcal{R}_{y_j}(\mathcal{E}_j)$, such that the cardinality of each repair set is at most $r_{l_j}$.
5:      Set $\mathcal{S}(\mathcal{E}_j, y_j) = \mathcal{E}_j \cup \mathcal{R}_1(\mathcal{E}_j) \cup \mathcal{R}_2(\mathcal{E}_j) \cup \cdots \cup \mathcal{R}_{y_j}(\mathcal{E}_j)$.
6:      if $H(\mathbf{Y}_{\mathcal{I}_{j-1} \cup \mathcal{S}(\mathcal{E}_j, y_j)}) < k$ then
7:          $\mathcal{I}_j = \mathcal{I}_{j-1} \cup \mathcal{S}(\mathcal{E}_j, y_j)$.
8:      end if
9:      Increment $j$.
10: end while
11: Set $z = j - 1$.
12: Pick $\mathcal{T}_z \subset [n] \setminus \mathcal{I}_z$ such that $|\mathcal{T}_z| = \sum_{m=1}^{z} (r_{l_m} y_m + l_m) - |\mathcal{I}_z|$ and $H(\mathbf{Y}_{\mathcal{I}_z \cup \mathcal{T}_z}) < k$.
13: Set $\mathcal{I} = \mathcal{I}_z \cup \mathcal{T}_z$.
```

As a result of the algorithm, the size of $\mathcal{I}$ finally becomes $\sum_{j=1}^{z} (r_{l_j} y_j + l_j)$. To show $H(\mathbf{Y}_{\mathcal{I}}) \leqslant A(\boldsymbol{l}, \mathbf{y})$, we first define a set Ψ_j as follows:

$$\Psi_j = \cup_{m=1}^{y_j - 1} \psi_m \cup \mathcal{E}_j,$$

where $\psi_m \in \mathcal{R}_m(\mathcal{E}_j) \setminus \mathcal{I}_{j-1}$ for $1 \leqslant j \leqslant z$. We can always find such element ψ_m by the selection rule in step 3-4. Then, we have

$$\begin{aligned} H(\mathbf{Y}_{\mathcal{I}}) = H(\mathbf{Y}_{\mathcal{I} \setminus \cup_{j=1}^{z} \Psi_j}) &\leqslant \left| \mathcal{I} \setminus \cup_{j=1}^{z} \Psi_j \right| \\ &\overset{(a)}{=} |\mathcal{I}| - \sum_{j=1}^{z} |\Psi_j| \\ &= \sum_{j=1}^{z} (r_{l_j} y_j + l_j) - \sum_{j=1}^{z} (l_j + (y_j - 1)) \\ &= \sum_{j=1}^{z} ((r_{l_j} - 1) y_j + 1) = A(\boldsymbol{l}, \mathbf{y}) < k. \end{aligned} \tag{5.2}$$

We note that (a) follows from $\Psi_j \cap \mathcal{I}_{j-1} = \emptyset$ for $1 \leqslant j \leqslant z$, and thus, $\Psi_1, \Psi_2, \ldots, \Psi_z$ are disjoint.

Now, it is sufficient to show that, for such a set $\mathcal{I}$, there always exists an $(n-|\mathcal{I}|, k-A(\boldsymbol{l}, \mathbf{y}), d)_q$ code. Consider a codeword $\mathbf{Z} \in \mathcal{Z} \triangleq \{\mathbf{Y}_{\mathcal{I}} : \mathbf{Y} \in \mathcal{C}\}$. We denote a codebook $\tilde{\mathcal{C}}(\mathbf{Z}) = \{\mathbf{Y}_{[n]\backslash\mathcal{I}} : \mathbf{Y}_{\mathcal{I}} = \mathbf{Z}\}$. The length of a codeword in $\tilde{\mathcal{C}}(\mathbf{Z})$ is $n - |\mathcal{I}|$. It is not difficult to show that codewords in $\tilde{\mathcal{C}}(\mathbf{Z})$ have the minimum distance at least d. We first show that there exists at least one $\mathbf{Z} \in \mathcal{Z}$ such that $|\tilde{\mathcal{C}}(\mathbf{Z})| \geqslant q^{k-A(\boldsymbol{l},\mathbf{y})}$. Assume on the contrary that $|\tilde{\mathcal{C}}(\mathbf{Z})| < q^{k-A(\boldsymbol{l},\mathbf{y})}$ for every $\mathbf{Z} \in \mathcal{Z}$. Then, we have

$$\begin{aligned}\sum_{\mathbf{Z}\in\mathcal{Z}} |\tilde{\mathcal{C}}(\mathbf{Z})| &= |\mathcal{C}| = q^k < |\mathcal{Z}| \cdot q^{k-A(\boldsymbol{l},\mathbf{y})} \\ &= q^{H(\mathbf{Y}_{\mathcal{I}})} \cdot q^{k-A(\boldsymbol{l},\mathbf{y})} = q^{k+(H(\mathbf{Y}_{\mathcal{I}})-A(\boldsymbol{l},\mathbf{y}))}. \qquad (5.3)\end{aligned}$$

Since $H(\mathbf{Y}_{\mathcal{I}}) - A(\boldsymbol{l},\mathbf{y}) \leqslant 0$ by (5.2), the equation (5.3) violates the assumption. Thus, there is at least one $\mathbf{Z} \in \mathcal{Z}$ such that $|\tilde{C}(\mathbf{Z})| \geqslant q^{k-A(\boldsymbol{l},\mathbf{y})}$ thereby resulting in an $(n - |\mathcal{I}|, k - A(\boldsymbol{l},\mathbf{y}), d)_q$ code. Since, for all the possible $\boldsymbol{l}$ and $\mathbf{y}$, $k - A(\boldsymbol{l},\mathbf{y})$ is less than or equal to $k_{opt}^{(q)}(n - |\mathcal{I}|, d)$, we obtain

$$k \leqslant \min_{\substack{z\in\mathbb{Z}^+ \\ \boldsymbol{l}=\{l_j\in[d-1],1\leqslant j\leqslant z\} \\ \mathbf{y}=\{y_j\in[t_{l_j}],1\leqslant j\leqslant z\} \\ A(\boldsymbol{l},\mathbf{y})<k}} \left[A(\boldsymbol{l},\mathbf{y}) + k_{opt}^{(q)}(n - |\mathcal{I}|,\ d)\right].$$

By defining $B(\boldsymbol{l},\mathbf{y}) = |\mathcal{I}|$, we complete the proof. ■

Theorem 11 For an $(n, k, d)_q$ code $\mathcal{C}$ that has joint information availability $\{(r_l, t_l) : l \in [d-1]\}$, we have

$$d \leqslant \min_{\substack{z\in\mathbb{Z}^+ \\ \boldsymbol{l}=\{l_j\in[d-1],1\leqslant j\leqslant z\} \\ \mathbf{y}=\{y_j\in[t_{l_j}],1\leqslant j\leqslant z\} \\ A(\boldsymbol{l},\mathbf{y})<k}} \left[d_{opt}^{(q)}(n - B(\boldsymbol{l},\mathbf{y}), k - A(\boldsymbol{l},\mathbf{y}))\right], \qquad (5.4)$$

where

$$A(\boldsymbol{l},\mathbf{y}) = \sum_{j=1}^{z}((r_{l_j} - 1)y_j + 1), \quad B(\boldsymbol{l},\mathbf{y}) = \sum_{j=1}^{z}(r_{l_j} y_j + l_j),$$

and $d_{opt}^{(q)}(n', k')$ is the largest possible minimum distance of a q-ary code of length n' and dimension k'.

Proof: Recall the proof of Theorem 10. For any positive integer z satisfying $A(\boldsymbol{l},\mathbf{y}) < k$, there always exists a set $\mathcal{I}$ such that $|\mathcal{I}| = B(\boldsymbol{l},\mathbf{y})$ and $H(\mathbf{Y}_{\mathcal{I}}) \leqslant A(\boldsymbol{l},\mathbf{y})$. For such a set $\mathcal{I}$, consider a codeword $\mathbf{Z} \in \mathcal{Z} \triangleq \{\mathbf{Y}_{\mathcal{I}} : \mathbf{Y} \in \mathcal{C}\}$ and the corresponding codebook $\tilde{\mathcal{C}}(\mathbf{Z}) = \{\mathbf{Y}_{[n]\setminus\mathcal{I}} : \mathbf{Y}_{\mathcal{I}} = \mathbf{Z}\}$. Let $\tilde{d}(\mathbf{Z})$ be the minimum distance of codewords in $\tilde{\mathcal{C}}(\mathbf{Z})$. Then, we have

$$d \leqslant \tilde{d}(\mathbf{Z}) \leqslant d_{opt}^{(q)}\left(n - |\mathcal{I}|,\ k - H(\mathbf{Y}_{\mathcal{I}})\right).$$

Since, for all the possible $\boldsymbol{l}$ and $\mathbf{y}$, $H(\mathbf{Y}_{\mathcal{I}}) \leqslant A(\boldsymbol{l},\mathbf{y})$, we obtain

$$d \leqslant \min_{\substack{z\in\mathbb{Z}^+ \\ \boldsymbol{l}=\{l_j\in[d-1],1\leqslant j\leqslant z\} \\ \mathbf{y}=\{y_j\in[t_{l_j}],1\leqslant j\leqslant z\} \\ A(\boldsymbol{l},\mathbf{y})<k}} \left[d_{opt}^{(q)}\left(n - |\mathcal{I}|,\ k - A(\boldsymbol{l},\mathbf{y})\right)\right],$$

By defining $B(\boldsymbol{l},\mathbf{y}) = |\mathcal{I}|$, we complete the proof. ■

Corollary 1 For an $(n, k, d)_q$ code $\mathcal{C}$ that has information availability (r_1, t_1), we have

$$k \leqslant \min_{\substack{z\in\mathbb{Z}^+ \\ \mathbf{y}=\{y_j\in[t_1],1\leqslant j\leqslant z\} \\ A(\mathbf{y})<k}} \left[A(\mathbf{y}) + k_{opt}^{(q)}\left(n - B(\mathbf{y}),\ d\right)\right], \tag{5.5}$$

$$d \leqslant \min_{\substack{z\in\mathbb{Z}^+ \\ \mathbf{y}=\{y_j\in[t_1],1\leqslant j\leqslant z\} \\ A(\mathbf{y})<k}} \left[d_{opt}^{(q)}\left(n - B(\mathbf{y}), k - A(\mathbf{y})\right)\right], \tag{5.6}$$

where

$$A(\boldsymbol{y}) = \sum_{j=1}^{z}((r_1-1)y_j+1), \quad B(\boldsymbol{y}) = \sum_{j=1}^{z}(r_1 y_j + 1).$$

Proof: Since it is the case of $\boldsymbol{l} = (1,1,\ldots,1)$ in Theorem 10-11, we obtain (5.5) and (5.6) straightforwardly. ■

Corollary 2 For an $(n,k,d)_q$ code $\mathcal{C}$ that has joint information locality $\{r_l : l \in [d{-}1]\}$, we have

$$k \leqslant \min_{\substack{z\in\mathbb{Z}^+ \\ \boldsymbol{l}=\{l_j\in[d-1],1\leqslant j\leqslant z\} \\ A(\boldsymbol{l})<k}} \left[A(\boldsymbol{l}) + k_{opt}^{(q)}(n-B(\boldsymbol{l}),\ d)\right], \tag{5.7}$$

$$d \leqslant \min_{\substack{z\in\mathbb{Z}^+ \\ \boldsymbol{l}=\{l_j\in[d-1],1\leqslant j\leqslant z\} \\ A(\boldsymbol{l})<k}} \left[d_{opt}^{(q)}(n-B(\boldsymbol{l}), k-A(\boldsymbol{l}))\right], \tag{5.8}$$

where

$$A(\boldsymbol{l}) = \sum_{j=1}^{z} r_{l_j}, \quad B(\boldsymbol{l}) = \sum_{j=1}^{z}(r_{l_j} + l_j).$$

Proof: Since it is the case of $\boldsymbol{y} = (1,1,\ldots,1)$ in Theorem 10-11, we obtain (5.7) and (5.8) straightforwardly. ■

Corollary 3 For an $(n,k,d)_q$ code $\mathcal{C}$ that has 1-information locality r_1, we have

$$k \leqslant \min_{z\in[\lceil \frac{k}{r_1}\rceil-1]} \left[zr_1 + k_{opt}^{(q)}(n-z(r_1+1),\ d)\right], \tag{5.9}$$

$$d \leqslant \min_{z\in[\lceil \frac{k}{r_1}\rceil-1]} \left[d_{opt}^{(q)}(n-z(r_1+1), k-zr_1)\right]. \tag{5.10}$$

Proof: It is the case of $\boldsymbol{l} = (1,1,\ldots,1)$ and $\boldsymbol{y} = (1,1,\ldots,1)$ in Theorem 10-11. From the condition $A(\boldsymbol{l},\boldsymbol{y}) < k$, we have $z \leqslant \lceil \frac{k}{r_1} \rceil - 1$. Then, we obtain (5.9) and (5.10).

■

We note that the bounds in Corollary 1-3 were also derived in [4,14,17] respectively. However, the authors in [17] considered only linear codes, and the authors in both [14] and [4] considered only LRCs with all symbol locality.

Now, we extend the bound in Theorem 11 into a bound without the alphabet constraint and also to some well-known Singleton-like bounds for LRCs.

Corollary 4 For an $(n,k,d)_q$ code $\mathcal{C}$ that has joint information availability $\{(r_l,t_l) : l \in [d-1]\}$, we have

$$d \leqslant n-k+1-\max_{\substack{z\in\mathbb{Z}^+ \\ \boldsymbol{l}=\{l_j\in[d-1],1\leqslant j\leqslant z\} \\ \boldsymbol{y}=\{y_j\in[t_{l_j}],1\leqslant j\leqslant z\} \\ A(\boldsymbol{l},\boldsymbol{y})<k}} \sum_{j=1}^{z}(l_j+(y_j-1)), \tag{5.11}$$

where $A(\boldsymbol{l},\boldsymbol{y})=\sum_{j=1}^{z}((r_{l_j}-1)y_j+1)$.

Proof: By using Singleton bound [1] for $d_{opt}^{(q)}(\cdot)$ in Theorem 11, we obtain the bound (5.11) straightforwardly. ■

Corollary 5 For an $(n,k,d)_q$ code $\mathcal{C}$ that has information availability (r_1,t_1), we have

$$d \leqslant n-k-\left\lceil \frac{(k-1)t_1+1}{(r_1-1)t_1+1} \right\rceil +2. \tag{5.12}$$

Proof: We validate the bound by means of some suitable choices of z, $\boldsymbol{l}$, and $\boldsymbol{y}$ in Corollary 4. For $r_1=1$, we use $z=k-1$, $\boldsymbol{l}=(1,1,\ldots,1)$, and $\boldsymbol{y}=(t_1,t_1,\ldots,t_1)$. For $t_1=1$, we use $z=\lceil\frac{k}{r_1}\rceil-1$, $\boldsymbol{l}=(1,1,\ldots,1)$, and $\boldsymbol{y}=(1,1,\ldots,1)$. For $r_1,t_1\geqslant 2$, we follow similar steps to the proof in [17]. For two non-negative integers x and α, let $k=x((r_1-1)t_1+1)+\alpha$, for $1\leqslant\alpha\leqslant(r_1-1)t_1+1$. We consider two cases: (1) $1\leqslant\alpha\leqslant r_1$. (2) $r_1<\alpha\leqslant(r_1-1)t_1+1$. For the first case, we choose $z=\lceil\frac{k}{(r_1-1)t_1+1}\rceil-1$, $\boldsymbol{l}=$

$(1,1,\ldots,1)$, and $\boldsymbol{y} = (t_1, t_1, \ldots, t_1)$. For the second case, we choose $z = \lceil \frac{k}{(r_1-1)t_1+1} \rceil$, $\boldsymbol{l} = (1,1,\ldots,1)$, and $\boldsymbol{y} = (y_1, y_2, \ldots, y_{z-1}, y_z) = (t_1, t_1, \ldots, t_1, \lceil \frac{\alpha-1}{r_1-1} \rceil - 1)$. In both cases, it is easy to check $A(\boldsymbol{l}, \boldsymbol{y}) < k$. Now, it is enough to show that

$$\sum_{j=1}^{z} (l_j + (y_j - 1)) \geqslant \left\lceil \frac{(k-1)t_1 + 1}{(r_1 - 1)t_1 + 1} \right\rceil - 1.$$

Case (1): For $1 \leqslant \alpha \leqslant r_1$, with the chosen z, $\boldsymbol{l}$, and $\boldsymbol{y}$, we have $k = z((r_1 - 1)t_1 + 1) + \alpha$. Then,

$$\begin{aligned}\sum_{j=1}^{z} (l_j + (y_j - 1)) &= zt_1 = \frac{(k-\alpha)t_1}{(r_1-1)t_1+1} \\ &= \frac{(k-1)t_1 + 1 + (r_1 - \alpha)t_1}{(r_1-1)t_1+1} - 1 \geqslant \left\lceil \frac{(k-1)t_1+1}{(r_1-1)t_1+1} \right\rceil - 1.\end{aligned}$$

Case (2): For $r_1 < \alpha \leqslant (r_1 - 1)t_1 + 1$, with the chosen z, $\boldsymbol{l}$, and $\boldsymbol{y}$, we have $k = (z-1)((r_1-1)t_1+1) + \alpha$. Then,

$$\begin{aligned}&\sum_{j=1}^{z} (l_j + (y_j - 1)) = (z-1)t_1 + \left\lceil \frac{\alpha - 1}{r_1 - 1} \right\rceil - 1 \\ &\geqslant (z-1)t_1 + \left\lceil \frac{(\alpha-1)t_1+1}{(r_1-1)t_1+1} \right\rceil - 1 = \left\lceil \frac{(k-1)t_1+1}{(r_1-1)t_1+1} \right\rceil - 1.\end{aligned}$$

■

The bound (5.12) was also derived in [16, 17] for only linear codes and in [15] for both linear and non-linear codes, but with an alternative proof.

Corollary 6 For an $(n, k, d)_q$ code $\mathcal{C}$ that has l-information locality r_l, we have

$$d \leqslant n - k + 1 - l\left(\left\lceil \frac{k}{r_l} \right\rceil - 1\right). \tag{5.13}$$

Proof: It is the case of $\boldsymbol{l} = (l, l, \ldots, l)$ and $\boldsymbol{y} = (1, 1, \ldots, 1)$ in Corollary 4. From the condition $A(\boldsymbol{l}, \boldsymbol{y}) < k$, we have $z \leqslant \lceil \frac{k}{r_l} \rceil - 1$. Then, we obtain the bound (5.13). ■

The bound (5.13) was also derived in [11, 14] for LRCs with all symbol locality.

LRCs with information availability can be viewed as systematic batch codes with restricted query size [36]. Based on the connection between such batch codes and LRCs, we derive a well-known bound in [36] for batch codes using our bound in Corollary 4. First, recall the definition of the batch codes.

Definition 17 [36] A primitive (k, n, γ, τ) batch code $\mathcal{C}$ with restricted query size over alphabet $\mathcal{Q}$ encodes a string $\mathbf{X} \in \mathcal{Q}^k$ into a string $\mathbf{Y} \in \mathcal{Q}^n$, such that for all multisets of indices $i_1, i_2, \ldots, i_\tau$, where all $i_j \in [k]$, each of the entries $X_{i_1}, X_{i_2}, \ldots, X_{i_\tau}$ can be retrieved independently of each other by reading at most γ symbols of $\mathbf{Y}$. It is assumed that the symbols used to retrieve each of the variables X_{i_j}, for $1 \leqslant j \leqslant \tau$, are all disjoint.

□

Corollary 7 Let $\mathcal{C}$ be a systematic primitive (k, n, γ, τ) batch code. Then, the minimum distance d of $\mathcal{C}$ satisfies

$$d \leqslant n - k - (\tau - 1)\left(\left\lceil \frac{k}{\gamma\tau - \tau + 1} \right\rceil - 1\right) + 1. \tag{5.14}$$

Proof: In Corollary 4, we use $z = \lceil \frac{k}{(r_1-1)t_1+r_1} \rceil - 1$, $\boldsymbol{l} = (1, 1, \ldots, 1)$, and $\boldsymbol{y} = (t_1, t_1, \ldots, t_1)$. Then,

$$\begin{aligned} A(\boldsymbol{l}, \boldsymbol{y}) &= \left(\left\lceil \frac{k}{(r_1-1)t_1 + r_1} \right\rceil - 1\right)((r_1-1)t_1 + 1) \\ &< \left(\frac{k}{(r_1-1)t_1 + 1}\right)((r_1-1)t_1 + 1) = k. \end{aligned}$$

Since $\sum_{j=1}^{z}(l_j + (y_j - 1)) = zt_1 = t_1\left(\lceil \frac{k}{(r_1-1)t_1+r_1} \rceil - 1\right)$,

$$d \leqslant n - k - t_1\left(\left\lceil \frac{k}{r_1(t_1+1) - (t_1+1) + 1} \right\rceil - 1\right) + 1.$$

From the relation between an $(n, k, d)_q$ LRC with information availability (r_1, t_1) and a systematic (k, n, γ, τ) batch code, we finally obtain the bound (5.14) by setting $r_1 = \gamma$ and $t_1 = \tau - 1$. ■

We note that an alternative proof of the bound (5.14) was shown in [36] for both systematic and non-systematic codes (but only for linear codes).

5.3 Achievability and Tightness of the Proposed Bounds

Since existing bounds in [4, 14, 17] are special cases of the bounds (5.1) and (5.4), all the optimal codes in terms of the bounds in [4, 14, 17] also achieve the bounds (5.1) and (5.4) with equality.

In Table 5.1, we demonstrate the optimality of Combination codes and punctured Combination codes with certain choice of parameters in terms of the bounds in [4,14,17] and the proposed bounds (5.1) and (5.4). To obtain the locality and availability properties of punctured Combination codes, we carried out a computer simulation. We also use the online table [42] for $k^{(2)}_{opt}(\cdot, \cdot)$ and $d^{(2)}_{opt}(\cdot, \cdot)$ in the bounds. The Δ_k represents the difference between the code dimension of the proposed code and the optimal value in terms of each bound. The Δ_d represents the difference between the minimum distance of the proposed code and the optimal value in terms of each bound. In the table, the codes with parameters $[7, 3, 4]_2$, $[15, 4, 8]_2$, and $[31, 5, 16]_2$ are binary Simplex codes, and the codes with parameters $[6, 3, 3]_2$, $[10, 4, 4]_2$, $[14, 4, 7]_2$, $[15, 5, 5]_2$, $[25, 5, 11]_2$, and $[30, 5, 15]_2$ are Combination codes. The rest of the codes in the table are punctured Combination codes obtained by using Algorithm 1. Since Algorithm 1 is a heuristic puncturing method, there are non-zero values of Δ_k and Δ_d for punctured Combination codes. We note that, it is open whether optimal codes in terms of the bounds always exist or not. Moreover, ours are the best known codes in terms of the bounds.

Table 5.1: Optimality of Combination codes and punctured Combination codes with regard to well-known alphabet-dependent bounds and the proposed bounds

$[n,k,d]_2$	w	p	$(r_1,r_2)_i$	$(t_1,t_2)_i$	$(r_1,r_2)_a$	$(t_1,t_2)_a$	Δ_k				Δ_d			
							[4]	[14]	[17]	(5.1)	[4]	[14]	[17]	(5.4)
$[6,3,3]$	2	0	$(2,3)$	$(2,1)$	$(2,3)$	$(2,1)$	0	0	0	0	0	0	0	0
$[7,3,4]$	3	0	$(2,3)$	$(3,1)$	$(2,3)$	$(3,1)$	0	0	0	0	0	0	0	0
$[8,4,3]$	2	2	$(2,3)$	$(2,1)$	$(2,4)$	$(1,1)$	0	0	0	0	0	0	0	0
$[9,4,3]$	2	1	$(2,3)$	$(2,2)$	$(2,3)$	$(2,1)$	1	0	0	0	1	1	1	1
$[10,4,4]$	2	0	$(2,3)$	$(3,2)$	$(2,3)$	$(3,1)$	1	0	0	0	0	0	0	0
$[11,4,4]$	3	3	$(2,3)$	$(3,2)$	$(2,3)$	$(3,2)$	1	1	1	1	1	1	1	1
$[12,4,5]$	3	2	$(2,3)$	$(4,2)$	$(2,3)$	$(4,2)$	0	0	0	0	1	1	1	1
$[13,4,6]$	3	1	$(2,3)$	$(5,3)$	$(2,3)$	$(5,2)$	0	0	0	0	0	0	0	0
$[14,4,7]$	3	0	$(2,3)$	$(6,3)$	$(2,3)$	$(6,3)$	0	0	0	0	0	0	0	0
$[15,4,8]$	4	0	$(2,3)$	$(7,3)$	$(2,3)$	$(7,3)$	0	0	0	0	0	0	0	0
$[10,5,3]$	2	5	$(2,3)$	$(2,1)$	$(2,4)$	$(1,1)$	0	0	0	0	1	1	0	0
$[11,5,3]$	2	4	$(2,3)$	$(2,1)$	$(2,4)$	$(1,1)$	1	1	1	1	1	1	1	1
$[12,5,3]$	2	3	$(2,3)$	$(2,1)$	$(2,4)$	$(1,1)$	2	2	1	1	1	1	1	1
$[13,5,4]$	2	2	$(2,3)$	$(3,2)$	$(2,4)$	$(2,1)$	2	2	1	1	1	1	1	1
$[14,5,4]$	2	1	$(2,3)$	$(3,3)$	$(2,3)$	$(3,1)$	2	2	2	2	2	2	2	2
$[15,5,5]$	2	0	$(2,3)$	$(4,3)$	$(2,3)$	$(4,1)$	1	1	1	1	1	1	1	1
$[16,5,5]$	3	9	$(2,3)$	$(4,3)$	$(2,4)$	$(3,1)$	2	2	1	1	2	2	2	2
$[17,5,6]$	3	8	$(2,3)$	$(5,3)$	$(2,3)$	$(3,1)$	2	2	1	1	2	2	2	2
$[18,5,6]$	3	7	$(2,3)$	$(5,3)$	$(2,3)$	$(3,2)$	3	2	2	2	2	2	2	2
$[19,5,7]$	3	6	$(2,3)$	$(6,4)$	$(2,3)$	$(4,2)$	2	2	1	1	1	1	1	1
$[20,5,8]$	3	5	$(2,3)$	$(7,4)$	$(2,3)$	$(5,2)$	2	2	1	1	1	1	1	1
$[21,5,8]$	3	4	$(2,3)$	$(7,4)$	$(2,3)$	$(6,3)$	3	3	2	2	2	2	2	2
$[22,5,9]$	3	3	$(2,3)$	$(8,4)$	$(2,3)$	$(6,3)$	1	1	1	1	1	1	1	1
$[23,5,9]$	3	2	$(2,3)$	$(8,4)$	$(2,3)$	$(7,4)$	2	1	1	1	2	2	2	2
$[24,5,10]$	3	1	$(2,3)$	$(9,5)$	$(2,3)$	$(8,4)$	2	1	1	1	2	2	2	2
$[25,5,11]$	3	0	$(2,3)$	$(10,6)$	$(2,3)$	$(9,5)$	1	1	1	1	1	1	1	1
$[26,5,11]$	4	4	$(2,3)$	$(10,6)$	$(2,3)$	$(10,5)$	2	1	1	1	1	1	1	1
$[27,5,12]$	4	3	$(2,3)$	$(11,6)$	$(2,3)$	$(11,5)$	2	1	1	1	1	1	1	1
$[28,5,13]$	4	2	$(2,3)$	$(12,6)$	$(2,3)$	$(12,6)$	0	0	0	0	1	1	1	1
$[29,5,14]$	4	1	$(2,3)$	$(13,6)$	$(2,3)$	$(13,6)$	0	0	0	0	0	0	0	0
$[30,5,15]$	4	0	$(2,3)$	$(14,7)$	$(2,3)$	$(14,7)$	0	0	0	0	0	0	0	0
$[31,5,16]$	5	0	$(2,3)$	$(15,7)$	$(2,3)$	$(15,7)$	0	0	0	0	0	0	0	0

Now, we show the achievability and tightness of the bound in Theorem 10 using graph-based LRCs in [12–14]. Based on the graph representation of the codes, we can easily obtain their joint information availability. With certain choice of parameters, we have some optimal and almost optimal LRCs in terms of our bound. Here, the almost optimality refers to having the value one less than the optimal case. The result is summarized in Table 5.2. To show the tightness, we use the parameters of $(32, 8, 7)_2$ complete multipartite graph code with joint information availability $\{(r_1, t_1), (r_2, t_2), (r_3, t_3)\} = \{(2, 6), (3, 5), (4, 5)\}$. With the parameters, we compare the optimal dimension $k_{b\text{-}opt}$ of our bound with those of the bounds in [4, 14, 17]. The bound in [4] gives $k_{b\text{-}opt} = 16$. The bounds in [14, 17] give $k_{b\text{-}opt} = 14$. Our bound gives $k_{b\text{-}opt} = 13$, and thus, it is the tightest.

Table 5.2: Optimality of graph-based LRCs with joint information availability

Graph-based LRCs	Joint information availability	Optimality
$\left(\frac{k(k+1)}{2},k,k\right)_2$ CG codes	$r_l=l+1,$ $t_l=k-l$	Optimal cases: $k=p=3,4,$ Almost optimal cases: $k=p=5$
$\left(\frac{k(k-\frac{k}{p}+2)}{2},k,k-\frac{k}{p}+1\right)_2$ CMG codes	$r_l=l+1,$ $t_1=k-\frac{k}{p},$ $t_2=k-\frac{2k}{p}+1,$ $t_3=\begin{cases}k-\frac{3k}{p}+3, & \text{for } p\geqslant 3,\\ 2, & \text{for } p=2.\end{cases}$	Optimal cases: $k=4,p=2,$ Almost optiaml cases: $k=6,p=3$
$(3k-3,k,3)_2$ Tiara codes	$(r_1,t_1)=(2,2),$ $(r_2,t_2)=(3,2)$	Optimal case: $k=4$
$(3k-5,k,3)_2$ Crown codes	$(r_1,t_1)=(2,2),$ $(r_2,t_2)=(3,1)$	Optimal case: $k=5$
$(2k,k,3)_2$ Ring codes	$(r_1,t_1)=(2,2),$ $(r_2,t_2)=(4,1)$	Optimal cases: $k=6,7,$ Almost optimal case: $k=8$

Chapter 6

Concluding Remarks and Future Directions

In this dissertation, we aimed at designing LRCs with good locality and availability properties. We focused on three main issues: (1) graph-based LRC construction, (2) puncturing-based LRC construction, (3) new alphabet-dependent bounds for LRCs.

First, we proposed a graph-based LRC construction using simple graphs to design binary LRCs with good joint locality. Then, we generalize the construction using hypergraphs. From the construction, if we have an (r, t)-hypergraph, we can construct a binary LRC with $(r, t)_i$-availability. The necessary conditions for the existence of an (r, t)-hypergraph were presented. For $r = 2$ and 3, the necessary conditions are also sufficient. For $r \geqslant 4$, we can always have at least one (r, t)-hypergraph with sufficiently large v. As a special case, we can construct binary LRCs with $(r, t)_a$-availability from (r, t)-tower hypergraphs. For given r and t, an (r, t)-tower hypergraph always exists. We also proposed extended hypergraph codes to increase the minimum distance. Interestingly, all the proposed codes achieve the bound (3.2) with equality.

Secondly, we proposed block-punctured Simplex codes, named Combination (k, w)

codes. The proposed codes are simply obtained by a process called block-puncturing that punctures all the columns of weights from k down to $w+1$, in the generator matrix of a binary Simplex code of dimension k. The minimum distance, joint locality $(r_1, r_2)_a$, joint information locality, availability, and joint information availability of the proposed codes are determined in closed-form expressions. To increase the code rate more, we also suggested another puncturing method, called subblock puncturing, that punctures few more columns of weight w from the Combination (k, w) code. As we have expected, the punctured Combination codes also have good joint locality and availability. Both of Combination codes and punctured Combination codes with certain choices of parameters attain well-known upper bounds on the code dimension and minimum distance which take into account the field size, locality, and availability. Moreover, our codes have the best code rate among the existing LRCs with the same availability.

Finally, we introduced generalized notions of the original availability, the availability for multiple symbols, joint availability, and joint information availability. Then, we proposed two new alphabet-dependent bounds for LRCs with joint information availability. Interestingly, it is possible to deduce some well-known alphabet-dependent bounds and Singleton-like bounds without the alphabet constraint. Thus, the proposed bounds provide a more general viewpoint that allows to derive some previous bounds from the same generic result.

There are several important questions that remain open. In Chapter 3, the existence problem of optimal LRCs for two classes 3) and 4) in Remark 1 for $r, t \geqslant 2$ remains open. Lots of cases for the case 2) are unknown yet. In Chapter 4, the tightness of the bound (4.6) and construction of optimal codes with respect to the bound also remain open. In Chapter 5, only some of optimal codes in terms of the proposed alphabet-dependent bounds (5.1) and (5.4) are found.

As a future work, one may obtain LRCs with more various parameters using new equivalence relations and useful graph properties. Addressing this would potentially lead to valuable contributions towards both graph theory and coding theory. To find a better puncturing algorithm than ours for LRCs is also an interesting and challenging task. As another future direction, one may consider the analysis for joint availability of existing codes, and another construction for new codes with good joint locality and availability.

Bibliography

[1] W. C. Huffman and V. Pless, “Fundamentals of error correcting codes,” *University Press, Cambridge*, 2003.

[2] P. Gopalan, C. Huang, H. Simitci, and S. Yekhanin, “On the locality of codeword symbols,” *IEEE Trans. Inf. Theory*, vol. 58, no. 11, pp. 6925-6934, Nov. 2012.

[3] M. Forbes and S. Yekhanin, “On the locality of codeword symbols in non-linear codes,” *Discrete Math.*, vol. 324, pp. 78-84, Jun. 2013.

[4] V. Cadambe and A. Mazumdar, “Bounds on the size of locally recoverable codes,” *IEEE Trans. Inf. Theory*, vol. 61, no. 11, pp. 5787-5794, Nov. 2015.

[5] D. S. Papailiopoulos and A. G. Dimakis, “Locally repairable codes,” *IEEE Trans. Inf. Theory*, vol. 60, no. 10, pp. 5843-5855, Oct. 2014.

[6] C. Huang, M. Chen, and J. Li, “Pyramid codes: Flexible schemes to trade space for access efficiency in reliable data storage systems,” *in Proc. IEEE Int. Symp. Netw. Comput. Appl.*, pp. 79-86, Jul. 2007.

[7] I. Tamo and A. Barg, “A family of optimal locally recoverable codes,” *IEEE Trans. Inf. Theory*, vol. 60, no. 8, pp. 4661-4676, Aug. 2014.

[8] I. Tamo, D. S. Papailiopoulos, and A. G. Dimakis, “Optimal locally repairable codes and connections to matroid theory,” *IEEE Trans. Inf. Theory*, vol. 62, no. 12, pp. 6661-6671, Dec. 2016.

[9] N. Prakash, G. M. Kamath, V. Lalitha, and P. V. Kumar, “Optimal linear codes with a local-error-correction property,” *In Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, pp. 2776-2780, Jul. 2012.

[10] W. Song, S. H. Dau, C. Yuen, and T. J. Li, “Optimal locally repairable linear codes,” *IEEE J. Sel. Areas Commun.*, vol. 32, no. 5, pp. 1019-1036, May 2014.

[11] A. S. Rawat, A. Mazumdar, and S. Vishwanath, “Cooperative local repair in distributed storage,” *EURASIP J. Adv. Signal Processing*, vol. 2015, no. 107, pp 1-17, Dec. 2015.

[12] J.-H. Kim, M.-Y. Nam, and H.-Y. Song, “Binary locally repairable codes from complete multipartite graphs,” *in Proc. Int. Conf. on ICT Convergence (ICTC)*, pp. 1093-1095, Oct. 2015.

[13] J.-H. Kim, M.-Y. Nam, K.-H. Park, and H.-Y. Song, “Optimal binary locally repairable codes with joint information locality,” *in Proc. IEEE Information Theory Workshop (ITW)*, pp. 54-58, Oct. 2015.

[14] J.-H. Kim, M.-Y. Nam, and H.-Y. Song, “Alphabet-dependent upper bounds for locally repairable codes with joint locality,” *In Proc. Int. Symp. Inf. Theory and Its Applications (ISITA)*, pp. 36-40, Oct. 2016.

[15] A. S. Rawat, D. S. Papailiopoulos, A. G. Dimakis, and S. Vishwanath, "Locality and availability in distributed storage," *IEEE Trans. Inf. Theory*, vol. 62, no. 8, pp. 4481-4493, Aug. 2016.

[16] A. Wang and Z. Zhang, "Repair locality with multiple erasure tolerance," *IEEE Trans. Inf. Theory*, vol. 60, no. 11, pp. 6979-6987, Nov. 2014.

[17] P. Huang, E. Yaakobi, H. Uchikawa, and P. H. Siegel, "Binary linear locally repairable codes," *IEEE Trans. Inf. Theory*, vol. 62, no. 11, pp. 6268-6283, Nov. 2016.

[18] M. Sathiamoorthy, M. Asteris, and D. Papailiopoulos, "XORing elephants: Novel erasure codes for big data," *Proc. VLDB Endowment*, vol. 6, no. 5, pp. 325-336, 2013.

[19] R. G. Gallager, *Low density parity-check codes*, Cambridge, MA, MIT Press, 1963.

[20] C. Berge, *Graphs and Hypergraphs*, North-Holland, 1973.

[21] J.-H. Kim and H.-Y. Song, "Hypergraph-based Binary Locally Repairable Codes with Availability," submitted to *IEEE Comm. Letters*, 2017.

[22] C. J. Colbourn and J. H. Dinitz, *Handbook of Combinatorial Designs*, CRC press, 2007.

[23] I. Tamo, Z. Wang, and J. Bruck, "Zigzag codes: MDS array codes with optimal rebuilding," *IEEE Trans. Inf. Theory*, vol. 59, no. 3, pp. 1597-1616, Mar. 2013.

[24] Y.-S. Su, "Design of membership matrices for (r, t)-availability in distributed storage," *in Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, pp. 998-1002, Jul. 2016.

[25] J. Hao, S.-T. Xia and Bin Chen, “Some results on optimal locally repairable codes,” *in Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, pp 440-444, Jul. 2016.

[26] J. Hao and S.-T. Xia, “Constructions of optimal binary locally repairable codes with multiple repair groups,” *IEEE Comm. Letters*, vol. 20, no. 6, pp. 1060-1063, Jun. 2016.

[27] Y. Kou, S. Lin, and M. P. C. Fossorier, “Low-density parity-check codes based on finite geometries: a rediscovery and new results,” *IEEE Trans. Inf. Theory*, vol. 47, no. 7, pp. 2711-2736, Nov. 2001.

[28] M. P. C. Fossorier, “Quasicyclic low-density parity-check codes from circulant permutation matrices,” *IEEE Trans. Inf. Theory*, vol. 50, no. 8, pp. 1788-1793, Aug. 2004.

[29] Y.-S. Su, “On constructions of a class of binary locally repairable codes with multiple repair groups,” *IEEE Access*, vol. 5, pp. 3524-3528, 2017.

[30] A. Wang, Z. Zhang, and M. Liu, “Achieving arbitrary locality and availability in binary codes,” *in Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, pp. 1866-1870, Jun. 2015.

[31] M. Bras-Amorós and K. Stokes, “The semigroup of combinatorial configurations,” *Semigroup Forum*, vol. 84, no. 1, pp. 91-96, Jan. 2012.

[32] J.-H. Kim, Mi-Young Nam, and H.-Y. Song, “Block-Punctured Binary Simplex Codes for Local and Parallel Repair in Distributed Storage Systems,” submitted to *IEEE Trans. Inf. Theory*, 2017.

[33] I. Dumer and O. Kapralova, “Spherically punctured biorthogonal codes,” *IEEE Trans. Inf. Theory*, vol. 59, no. 9, pp. 6010-6017, Sep. 2013.

[34] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai, “Batch codes and their applications,” *In Proc. 36th Annu. ACM Symp. Theory Comput. (STOC)*, pp. 262-271, Jun. 2004.

[35] Z. Wang, O. Shaked, Y. Cassuto, and J. Bruck, “Codes for network switches,” *In Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, pp. 1057-1061, Jul. 2013.

[36] H. Zhang and V. Skachek, “Bounds for batch codes with restricted query size,” *In Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, pp. 1192-1196, Jul. 2016.

[37] Z. Wang, H. M. Kiah, and Y. Cassuto, “Optimal binary switch codes with small query size,” *In Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, pp. 636-640, Jun. 2015.

[38] S. Lin and D. J. Costello, “Error control coding,” *Prentice Hall*, 2004.

[39] I. Tamo, A. Barg, and A. Frolov, “Bounds on the Parameters of locally recoverable codes,” *IEEE Trans. Inf. Theory*, vol. 62, no. 6, pp. 3070-3083, Jun. 2016.

[40] A. Wang and Z. Zhang, “Repair locality from a combinatorial perspective,” *In Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, pp. 1972-1976, Jun. 2014.

[41] J.-H. Kim and H.-Y. Song, “Alphabet-dependent Bounds for Locally Repairable Codes with Joint Information Availability,” *IEEE Comm. Letters*, vol. 21, no. 8, pp. 1-4, Aug. 2017.

[42] M. Grassl, “Code Tables: Bounds on the parameters of various types of codes,” *[Online]. Available: http://www.codetables.de/*, 2017.

국문요약

그래프 기반 부분접속복구 부호

본 논문은 임의의 복호 가능한 손실 심볼 집합에 대해 서로 소인 심볼 집합들 중 하나로 복구가 가능한 부호인 부분접속복구 부호에 대해 다룬다. 특히, 어떤 손실 심볼이 각각 최대 r개의 심볼들을 원소로 갖는 t개의 서로 소인 심볼 집합들 중 하나로 복구가 가능한 경우, 그 심볼의 부분접속수를 r이라고 하고 가용도를 t라고 한다.

첫 번째 주제로, 좋은 부분접속수와 가용도 특성을 갖는 새로운 그래프 기반 부분접속복구 부호를 제안한다. 이를 위해, 완전 그래프, 완전 다분할 그래프와 같이 잘 알려진 그래프들과 새롭게 정의된 티아라 그래프, 크라운 그래프, 링 그래프를 사용한다. 이러한 심플 그래프들로부터 좋은 결합 부분접속수를 갖는 이진 부분접속복구 부호를 설계한다. 또한, 선형 r-균일 t-정규 하이퍼그래프를 사용하여 이를 확장한다. 이러한 하이퍼그래프로부터 부분접속수 r, 가용도 t를 갖는 이진 부분접속복구 부호를 설계할 수 있다. 이렇게 설계된 부분접속복구 부호는 Rawat 등이 2016년에 제안한 Singleton 유사 한계식을 등호로 만족하는 최적 부호이다. 이 결과는, 만약 이러한 최적 부분접속복구 부호를 설계한다면 그에 대응되는 하이퍼그래프 또한 얻을 수 있게 됨을 시사한다. 따라서, 본 연구 결과는 부분접속복구 부호와 하이퍼그래프 간의 관계에 대한 새로운 관점을 제시한다.

두 번째 주제로, 그래프 기반 부분접속복구 부호를 확장한 블락 펑쳐드 심플렉스 부호를 제안한다. 본 논문에서는 이를 컴비네이션 부호라고 부른다. 컴비네이션 부호와 펑쳐드 컴비네이션 부호는 Tamo 등이 2016년에 제안한 한계식 측면에서 알려진 부호들 중 가장 좋은 부호이다.

마지막 주제로, 기존 가용도의 일반화된 개념인 다수 심볼을 위한 가용도와 결합 가용도를 새롭게 정의하고, 정보 심볼에 대한 결합 가용도를 고려한 부분접속복구 부호를 위한 두 개의 새로운 알파벳 기반 한계식을 제안한다. 첫 번째 한계식은 부호의 차원에 대한 한계식이고, 두 번째 한계식은 최소거리에 대한 한계식이다. 이들 한계식은 Cadambe 등이 2015년에 제안한 한계식, Kim 등이 2016년에 제안한 한계식, Huang 등이 2016년에 제안한 한계식을 일반화시킨 결과이다.

핵심되는 말 : 부분접속복구 부호, 그래프 기반 부호, 하이퍼그래프, 천공된 부호, 알파벳 의존 바운드.